Archive

The subscription magazine for *Archimedes* users

December '87

Vol. 1 No. 3

Price £1.20

Sound information

ADFS Beginner's Guide

How fast is *Archimedes*?

BBC File Transfer Kit

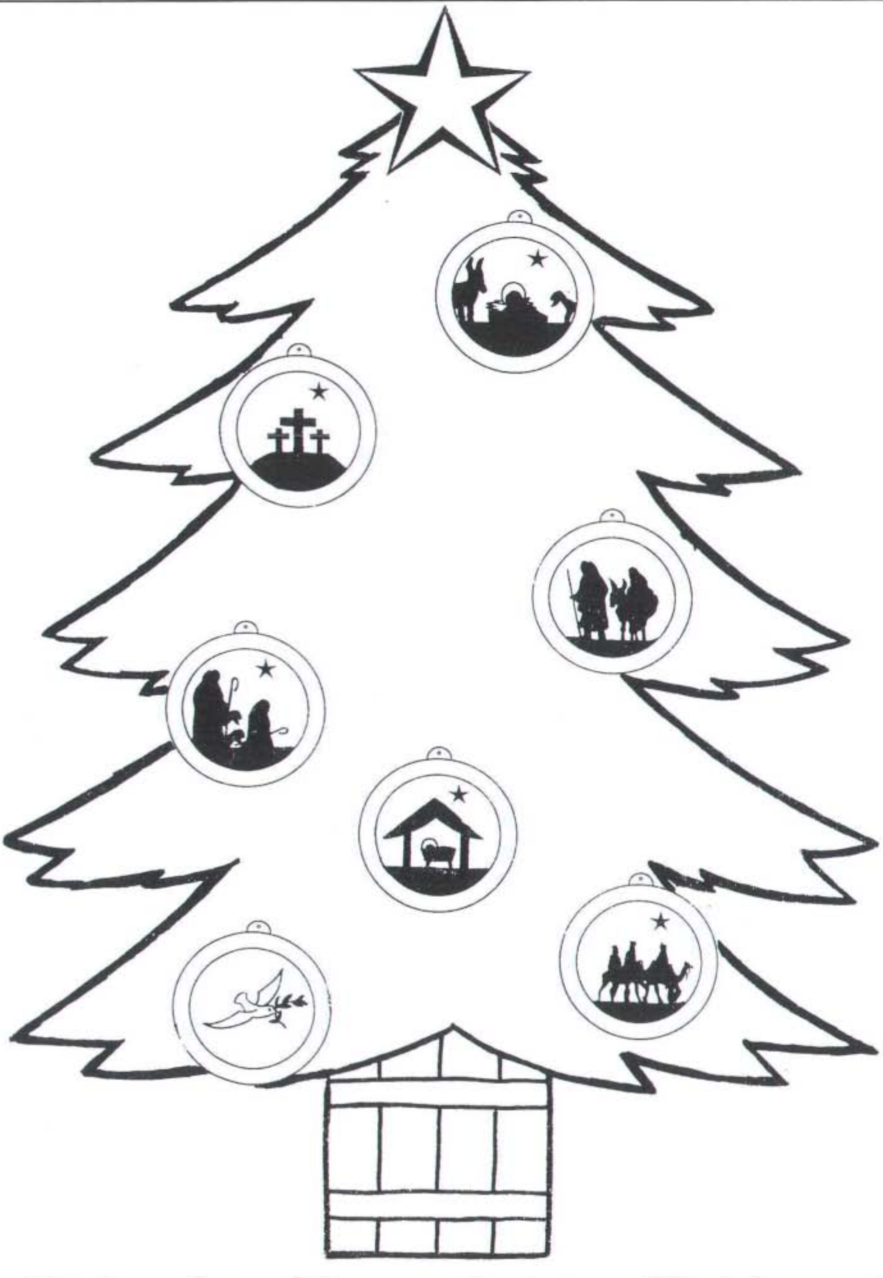
Colour Blender Program

More WIMP information

Plus:

Hints & Tips *Hints & Tips* *Hints & Tips* *Hints & Tips*
Hints & Tips *Hints & Tips* *Hints & Tips* *Hints & Tips*
Hints & Tips *Hints & Tips* *Hints & Tips* *Hints & Tips*

Archive is edited by Paul Beverley and published by Norwich Computer Services



May the true joy of the greatest ever Christmas gift
be yours this coming year!

Archive

Volume 1 • Nº 3 • December 1987

Contents

Archive goes on from strength to strength...

Once again, a very big thank you for all the support which readers have given us at Archive. The supply of information and articles which you send us has been tremendous. For example, in this issue there are a total of **seven pages of hints and tips!** If you want the standard of Archive to be maintained, please keep the information coming. If any of you would like to write review articles, let me know what sorts of things you are interested in.

We now have **well over 700 subscribers** and are still growing rapidly. It was good to meet a number of you at the Micro User Show and be able to put faces to the names. Over a 100 folk signed up at the Show. For those who did sign on then and didn't get a membership number, we've printed the numbers on the sticky address labels (if you haven't already thrown yours away!)

A special thank you to Tim Hooton, Mark Seeley and Matthew Treagus for their help in running the stand at the Micro User Show on the Friday and Saturday. I could not have managed without them. Sunday was not as busy, but being on a stand on your own all day has certain problems. I had to send an SOS message to Computer Concepts to get someone to man the stand for a few minutes so that I could get to the loo!

Apologies to Clive Williams who contributed the Mode 7 Screen Save Program last month. I did not put his name on the article. Sorry, Clive, but thanks for a useful routine and the informative notes that came with it.

Apologies, too, that I haven't managed to go through all the competition entries. I'll get that done in time for the next issue which should be out in the first week or ten days of January.

We are preparing a **Shareware Graphics Demonstration Disc** made up of some of the many programs that folk have sent in to Archive. There's only so much room in the magazine itself to put programs, so this seems a good way to get the programs out to those who want to use them.

We're also hoping to do **a disc of Mandelbrot patterns** – we've already got some really stunning ones – so if you have any that you have SCREENSAVEd and you would be prepared to donate for a shareware disc, please let us know. We can only get about ten or so on a disc, so we will have to select the best ones to include.

Thanks again for all your help.

Here's wishing you a Happy Christmas and a Prosperous New Year!

Paul Beverley

News, News, News......

Archimedes A400 series

• The A440 "should be a stock item by 1st/2nd December" but the A410, which is actually going to have a newly designed printed circuit board, won't be available until "probably early second quarter '88".

Programmers' Reference Guide

• The Reference Guide is ready at last (well, it should be by the time you receive this magazine!) but the price has gone up to £29.95 as it has had to be split into two books. There was physically too much material to bind into one volume. (Surely they should realised it would need two volumes since both the Master & the Compact have two part Reference Guides!)

We are just in the process of trying to organise a sensible discount on them so that we can offer them to members at less than £29 including post & packing. If you want to order one, I suggest you either ring us or send us a blank cheque marked "being not more than twenty-nine pounds" and we will fill it in.

Disc prices down!

• We can now supply branded discs (Wabash) for the same price that we were advertising the unbranded ones. Because of the increased circulation of Archive we were able to make a much bigger purchase of discs which has brought the price down accordingly, so 10 Wabash discs (DSDD) will now cost you only £16, though they do not come in a library box.

MS-DOS Prices up

• You will probably not be too pleased to hear (unless you have bought a 310M) that the price of the PC emulator with MS-DOS 3.21 has been increased to £99 plus VAT. The price of £89 we quoted last month was a "provisional price".

Music Think Tank

• During November Mike Beecher of Electro-Music Research apparently organised an "Archimedes Music Think Tank" at Acorn Computers headquarters in Cambridge. The aim was to try to coordinate the efforts of the various folk involved in developing hardware and software for using the Archimedes in any form of music generation, composing, recording etc. If you have any ideas of what you would like to see developed, get in touch with Mike.

5.25" Disc Interface

• Watford Electronics are developing a special interface to allow you to link your Archimedes up to one or more 5.25" drives. A number of people have got their 5.25" drives linked up directly to the Archimedes without any buffering, but Acorn say that by doing so, you risk blowing up the driver chips in the Archimedes. This interface should be available in a month or so. However, you should be aware that many of the 5.25" drives, especially the older ones are not really reliable enough for going up to the full 800k format.

Scottish User Group

• If you are interested in an **Archimedes User Group for Central Scotland**, contact the secretary, David Davidson, 2 Akarit Road, Larbert, Stirlingshire, FK5 4BY, telephone (0324) 558692. (I've also had a note from Barry Toft, 51 Arden Grove, Kilsyth, Glasgow who saying he is thinking about starting an AUG for Scotland, so I suggest they get in contact with each other if they have not already done so!)

1.2 Arthur

• The 'left hand' says that the new operating systems plus A-Writer won't be going out until the last week of December, but the 'right hand' told one of our readers that they were going out from 8th December.

...and Comments

What ever happened to the PC podule?...

At the Micro User Show earlier in the month there was a nasty rumour flying around that the development of Acorn's PC podule was "on hold". Informed sources within Acorn say that the company is not sure that having a PC podule that will allow control of a 5.25" drive and have CGA graphics will necessarily sell in great numbers. Also they are not sure whether it really will enhance the sales of the Archimedes itself enough to make it worth their while. By the middle of 1988, which is the earliest it would be available in any case, will it be what people really want?

The other consideration for Acorn is if someone buys an Archimedes with a PC podule and the dealer tells him that it is "PC compatible", the purchaser could quite easily turn round and ask where he plugs in his EGA board or some other piece of hardware. Perhaps we should be saying instead that the Archimedes "will run the majority of MS-DOS software", albeit at reduced speed under the emulator.

Yes, I know that we would like to be able to tell people that not only is Archimedes the fastest micro in the world (under £10,000) but that it is also IBM compatible, but is that necessarily the best way to break IBM's strangle-hold? I have great admiration for Acorn for choosing to go it alone – I wish them every success!

A raw deal for early Archimedes supporters...

As reported last month, Michael Page, Acorn's Corporate Communications Manager, said that the early Archimedes computers were for "mainly for software developers" and that the price originally quoted was "only a guide". But let us look at what early users have had to put up with.

- A price drop of £115
- PC emulator software costing £113.85 instead of £69 if purchased with the machine
- The mark I keyboard which leaves a lot to be desired compared with the mark II
- A provisional User Guide with no index*
- The 0.2 operating system and Welcome disc which are not upgradable to 0.3

I am afraid that when we get our 1.2 operating systems we will find that although the graphics routines are reputedly many times faster than earlier issues, there will still be bugs in it. However, I fear that Acorn will stick to their guns and say that this is the definitive version and that any new version of Arthur will have "extra facilities" and will therefore have to be a purchase not a free gift. I get a distinct feelings of dèja vu from early BBC days!

(*Acorn's comment in response is that with the new operating system we will be getting in mid-December, we will be getting a new Welcome Guide and a new User Guide whereas those who got the User Guides with an index will only be getting an up-date to the User Guide. This, of course, is in addition to a new Welcome disc and the WIMP-based A-Writer.)

Software, Hardware and Documentation

*We continue our list of what is available. This list is in addition to what was published in the first two issues. All the information about names, addresses and telephone numbers is in the **Fact-File** at the back of the magazine.*

• Those who are used to using our own **Continuous Processing ROM** with Wordwise Plus will be pleased to know that it is now available in disc form. The facilities are those of CP-ROM3, not CP-ROM2. We are making it available for £8 as an up-grade (i.e. if you already have CP-ROM 2 or 3) and as a new purchase for £18 including manual.

• The **Inter Series software** (-WORD -SHEET and -CHART) from Computer Concepts is now available but only on ROM, so you will need a back-plane and a ROM-podule before you can get started. CC are producing their own ROM podule (see below). If you already have any of the Inter-ROMs, you can upgrade them to the Archimedes versions for £11.50 per ROM but this can only be done directly through CC, not through the Archive magazine.

• **Computer Concepts' ROM podule** is to have 8 slots, like the Acorn equivalent, but instead of having only two slots for RAM and the other six slots for ROM, each of the eight slots can contain either RAM or ROM and there will be an optional battery back-up available. The unit will come with a ROM filing system so that ROM images can be loaded and saved and any RAM on the board can be used as a RAMdisc. Price: £49.95+VAT, (7.5% discount if bought through Archive.) availability "December" – probably around Christmas.

• **A true Archimedes word-processor!** Clares Micros have produced **Imagewriter**. Price £29.95 (£27.50 through the magazine) and it should be available by the time you receive this magazine. We will review it a.s.a.p.

• **APL Interpreter** – One of our readers tells us that a "free" APL interpreter will soon be available for the Archimedes. Anyone interested should get themselves put on the mailing list for the project, stating that they are using an Archimedes and, when it is available, they will be informed of the exact cost of disk and copying (of the order of a few pounds). They should write to:- I-APL Ltd, 2 Blenheim Road, St Albans, AL1 4NR. The interpreter is designed for educational use, but does follow the ISO standard.

• **Acorn Languages** – The full release versions of C, LISP and PROLOG are all available now and PASCAL and FORTRAN77 should be available by the time you get this magazine. A version of KERMIT is being tested at the moment and should be available at £49 + VAT at the beginning of January.

You need to be aware however that none of the compiled languages comes with an editor. Acornsoft supply an editor separately – TWIN which can also be used as a BASIC editor by using keywords TWIN or TWINO. The editor, which is used almost exclusively by Acorn's own software engineers, costs £29 plus VAT in its pre-release version, but Acorn have not yet decided what their up-grade policy is going to be when the full release version of TWIN becomes available. Acorn say that the present version is perfectly adequate for preparing source code files for the various compilers; it's just that the new version will make better use of the capabilities of the Archimedes.

• **Text processor/editor for compiled languages**. An alternative to TWIN is to be produced by Tubelink for 'probably' £19.95. Further details from Tubelink.

• **Advanced Statistical software** – An extremely impressive looking software package has been produced by Serious Statistical Software. The package contains so many different facilities, none of which means anything to me, that I won't even attempt to give a summary. If you know what "variation inflation factors" and the "Nelder-Mead algorithm" are then this package is probably for you! Price £150, but they will give 7.5% discount if you quote your Archive number. We are intending to do a full review of this in a later issue of Archive.

• **Contex Computing** have released a range of programs for the Archimedes. These are upgraded versions of the original BBC programs, making use of the superior speed and facilities of Archimedes. (Upgrades are available for existing users.) They include: **Bank Manager** (£25) for handling personal finance including a range of different accounts, credit accounts, transfers, cash flow, bank reconciliation, reports, foreign currency and lots more, **Business Utilities** (£12) which works with Manager and adds trials balance and spreadsheet analysis, suitable for self-employed businessmen and professionals, **Typing Tutor** (£15), **Easy Reader** (£12.50) (I'll have to get Tim to review it!) and **Madlibs** (£12.50), "a hilarious game which also teaches English grammar." Contex will give 10% discount to Archive readers if you quote your membership number when you order direct from them.

• **RESOURCE**, the educational software publishers are to publish a set of utilities to allow programmers to identify and cure problems in converting software to run under the 6502 emulator. For example, "**BBCrun**" creates an environment in which some "illegally" programmed software will operate as if on a standard BBC or Master. They are also doing a **Versatile Interface Podule** which will allow sampling of analogue data (up to 60kHz) via two A to D converters. 32 digital interface lines are also available on this system. To work with this is **EARS**, a dual channel digital recording system which can use the VIP to sample sounds and then play them back through the computer's audio system.

RESOURCE are also working on an up-date of their best selling "**DROOM**" – mathematical "adventure" type software for junior through to secondary pupils.

• **Another art package** has been published – **Arctist** from Fairhurst Instruments Ltd. – needs 1 Meg of ram but only costs £19.95. We'll try to get a comparative review done of that and Artisan.

• **...and a drawing package** – Pineapple Software have released their **DIAGRAM II** drawing package. It has been up-graded for the Archimedes. Again, we'll get this reviewed as soon as we can. A

New Competition?

I think the next competition ought to be to find the daftest error message that Archimedes puts out. So far we've had "Naff RTC month" (see last month's Q & A) and Damon Hoggett has discovered, in the Podule Module (hic, I mean sic!) "There ain't no loader to call" and, my favourite so far, "Bad stuff happen in da loader boss"! A

Hints and Tips

• **RAM upgrades** – If you want to up-grade the ram yourself, rather than trying to remove the main p.c.b. to get at the sockets where the new chips are to go, you can remove the front fascia. To do this, remove the lid, then disconnect the wires that go to the speaker and the 'power on' LED by pulling the four-pin socket off the pins on the board. The tongue at the front of this socket is a clip that holds the socket in place, so ease this forwards and the socket should slip off easily. Then you can remove the fascia itself by undoing the five screws, three underneath and one at each side. Care should be taken with the eject button on the disc drive as this can break off fairly easily if roughly handled. (Some say, will inevitably break off, but a bit of super-glue or the like is all you need to stick it back in place.) You will then have to remove the bridge that supports the disc drive(s). This can be done by removing one screw underneath the computer (do this first) then two screws at the side. Plug in the chips and reverse the process, again being careful of the disc eject button.

• **Fitting a second drive** is supposed to be a 'dealer only' upgrade, but as long as you know a bit about electronics and vaguely what happens inside computers, it is not too difficult to fit it yourself. However, you should watch out that the front fascia plate is not too high. If it is, it will bear on the disc and may cause an error when you try to access the disc. I discovered this when I found that the drives worked OK until I put the new two-hole fascia in place. My solution, when this happened, was "brute force and ignorance"! The metal bracket on which the drive is supported has a little bit of "give" in it – enough to raise the front of the drive by the couple of millimetres necessary to lift it away from the fascia.

The other important thing to know is that to configure your second drive as drive 1 you need to move the tiny black slider switch at the side of the drive. It is a four position switch and usually comes in position 0, so one click will move it to position 1.

• **Anti-Buzz Fix** – There are, apparently, two different buzzes. One is what occurs after you have pressed <escape> or <break> and the other a much more annoying buzz which not all computers seem to have. The first buzz is a software problem which is fixed in the 1.2 Arthur, so I am told, but the other requires a hardware fix which is supposed to be done by dealers as a free modification, but if you are deft with a soldering iron and are willing to risk your warranty, here's what to do…

The solution, according to Acorn's technical services department is…

"Solder a 100µF, 10 volt electrolytic capacitor across pins 7 (positive) and 4 (negative) of IC68. The capacitor should be kept as close to the p.c.b. as possible and should be secured to the board with glue or hot wax."

The bad news is that IC 68 is underneath the bar that supports the disc drive(s)! The easiest way to get at it is actually, (1) take off the lid, (2) unplug the lead that goes to the speaker and power-on LED (beware, the socket on the ends of the cable has a lip that locks it onto the pins on the board – ease the lip forward before trying to pull the socket off), (3) take off the front fascia (one screw at each side and three under the front edge) and (4) unscrew the drive-support bridge (one screw underneath and two at the side).

• **Problems with monitors** – Some folk are having problems with certain monitors. If the problem is lack of contrast, use an oscilloscope to check the voltage output levels from the Archimedes. If they are less than 0.7 volts peak-to-peak you may need to change the values of the output resistors. Acorn have changed resistors R20, R41 and R59 from 68 ohms to 43 ohms, so if you want to increase the output voltage, you could either change the resistors or solder a 120 ohm resistor in parallel with each.

The other problem with some monitors, especially the NEC and Fujitsu multisync monitors, is of getting a greenish tinge on white areas. This comes about because Acorn put the sync signal onto the green line which is apparently what certain monitors expect. To remove this sync signal, simply remove resistor R39 – a quick snip with a good pair of side-cutters should do it, but make sure you get the right resistor!

(When I tried to do these modifications, I found it wasn't too easy to decide which resistor was which because the numbers are actually underneath the resistors. If you look at the line of resistors coming away from the video output socket you will see that they are: R1, R3, R18, R20 (68R), R35, R37, R39 (1k2), R41 (68R), R45, R50, R52, R59 (68R), R60, R63 and R67.)

• **Archimedes on Econet** – As far as we can gather, the Econet hardware to be added to the basic 305 or 310 is just the same as the module which you would purchase for the Master or Compact – certainly, the part supplied by Acorn to one of our readers had the same part number as the Master equivalent. One problem which Econet users may find on earlier systems is that even if you only want to use the disc system, you still have to have a clock signal available, otherwise the computer hangs up! Presumably this will be corrected in the 1.2 operating system! If you are used to using !BOOT files on the network, you will need to change them all to !ARMBOOT as well as having !BOOT files for the BBC. The Archimedes will work quite well on a Level 3 server but there are no net utilities like VIEW, REMOTE, NOTIFY, ROFF etc. The only one provided is an enhanced FREE which includes RDFREE with it. Acorn have "no plans" for providing these utilities. This, for schools, is quite a problem as you cannot get to see what is going on around the net. However, software transfer around the net is very easy. (These comments were kindly provided by Mr V Smith of King Edwards School, Lytham.)

• **Control key abbreviations** – Lazy typists like me will like to know that if you want to type, say, MODE12 perhaps to list a program that was running in different screen mode or within a window, you can be abbreviate it to <ctrl-V><ctrl-L>. What you are doing is the equivalent of VDU22,12. On the BBC micro this was not a good idea because BASIC was unaware of the change of mode and would start to over-write screen memory with variables, but it is OK on the Archimedes because the screen memory is protected by having configured a certain amount of screen RAM. If you try it and then type PRINT MODE, it knows it is in mode 12. Other mode numbers can be worked out – mode 0 would be <cvtrl-V><ctrl-@>, 1 is A, 2-B, 3-C, 4-D, 5-E, 6-F, 7-G, 8-H, 9-I, 10-J, 11-K, 12-L, 13-M, 14-N, 15-O, 16-P, 17-Q, 18-R, 19-S, 20-T, 21-U.

This can be extended to things like changing background colour, say to blue, with <ctrl-S><ctrl-@><ctrl-D><ctrl-@><ctrl-@><ctrl-@> (where <ctrl-@> is actually done with <ctrl-shift-2>) but there comes a limit where it is quicker to type in the command rather than remembering the control codes. You can even do all the plotting functions in this way – try, for example, <ctrl-Y><e><ctrl-C><ctrl-C><ctrl-C><ctrl-C>. (That's a lower case "e", not <ctrl-e> so switch caps lock off and just press <e>. If nothing happens, do a mode change first to a graphics mode, say MODE 12, then try it.)

To change mode when in the Arthur Supervisor, you could use, say, ECHO || V || L or ECHO || S

|| @ || D || @ || @ || @ or you could use ECHO <19><0><4><0><0><0> but you can again just type in the <ctrl> sequences as mentioned above.

• **Special effects in View** – You can use *ECHO or use the control key sequences mentioned above when you are using View. Also, if you want to put the 'format block', 'move block' and 'delete block' commands onto function keys 10 to 12 (instead of using the <print> key) you can use:

*KEY 10 || ! || L

*KEY 11 || ! ||\

*KEY 12 || ! ,

Despite what it says in the User Guide about the pageup and pagedown keys not being used, they seem to work in View and they do actually move you a page at a time up and down.

• ***RMtidy** – Beware that on the 0.20 Arthur, this can cause the machine to crash whenm you subsequently try to us *RMLOAD.

• There is apparently another **undocumented screen mode** which will be available on Arthur 1.2 – mode 21 which is 640 x 512 in 256 colours, though it will obviously only be usable on a multi-sync monitor and uses 320k of RAM! Also, on the 400 series there will be two extra high resolution monochrome modes for 64kHz monitors – mode 22 which is 160 x 122 text with 1280 x 976 graphics and mode 23 which is text only at 144 x 54. These use the extra hardware that is on the 400 series boards though it looks as if there should be space for the chips on the 300 boards if you are prepared to risk fitting them yourself. There are no sockets, so you would have to solder-suck all the holes first, and it's a multi-layer board.

• **Delete on keypad** – If you compare the keypads of the Master and the Archimedes, you will see that where the Archimedes has a fullstop, the Master has a delete key. If you think it would be useful to have the delete function on the keypad, turn the num lock LED off and try pressing the fullstop key!

• **Function key definitions** – If you want to know what the current key definitions are, *SHOW K* will print them on the screen. The only slight confusion is that they appear in alphabetic order – KEY$0, KEY$1, KEY$10, KEY$11, KEY$12, KEY$13, KEY$14, KEY$15, KEY$2, KEY$3 etc!

Function key 0, as you probably know by now, is put onto the PRINT key, but where are the other function keys – 13, 14 and 15? The only one I have found is 13 which is on the INSERT key The other thing to watch is that although there is a separate key for function key 10 (the break key on the BBC micro), when you press <break>, KEY$10 is expanded as it was on the BBC micro! (That was on 0.2. Has it changed in later versions of the OS?)

• **Diary & notepad** – The diary and notepad can be saved onto disc by putting the pointer on the pad or the calendar and clicking the middle button. It then asks for confirmation that you want to save it. The notepad is saved as "notepad" and the diary as "Diary87" (or whatever year it is for). To load them back in again later, you have to open up the disc and click on the required file before clicking on the diary or notepad with the middle button and selecting LOAD. When saving, the name is fixed by the desktop program, but once it has been saved, you can rename the file if you want to save more than one, though obviously this applies more to the note-pad as I know that some of you are still having to use the notepad as a word-processor! Having said all that, I have to admit that when I was trying this out, I had problems saving the calendar – I kept getting "Disc full" or "Disc in need of compaction" errors. Any offers of explanation?

• **The SYSTEMDEVS module** is a set of logical device drivers that can be used from Arthur. They make the device appear to the programmer as if they were a file system. In Arthur 0.20 you have to load the module from the Welcome disc, but in 0.30 onwards it should be in ROM. They include LPT:, KEYBD:, PRINTER:, VDU and RAWVDU: so a simple command to copy a file to screen would be *COPY FILE VDU: and any non-printing characters appear in the format used to program the function keys, i.e. using pad characters so that, for example, ASCII 12 comes out as || L.

What is the point of these facilities? Well, you can use them to redirect the flow of data into or out of a program or relating to an Arthur command. Thus you can say `*EX {>info}` which sends the output from the EX command to a file called "info" and `*CAT {>>info}` will then ADD the catalogue information onto the end of the info file, or `*CAT {> PRINTER:}` would print out the catalogue. `*BASIC {<DATA} PROG` would run the BASIC program PROG and take it input information from the DATA file rather than from the keyboard. Another possible application is for debugging a program that is sending data to a disc file. Rather than stopping the program and examining the disc file periodically, you could change the line in the program where you set up the file for output and use instead `X% = OPENOUT "PRINT:"` then subsequent `PRINT#X%`'s or `BPUT#X%`'s would go to the printer. If the output is un-printable (or do I mean non-printing?!) characters then you could set the printer into a hex dump format which many dot matrix printers have these days. Then a final suggestion for an application would be when using network and ADFS. To avoid switching between the two, you could say `X%=OPENOUT "ADFS:$.TEST"`. This means that you could presumably have files open on both the network and the disk at the same time, but not having a network for my solitary Archimedes, I cannot check this!

(These comments were derived from an article in "Eureka" the Auckland BBC User Groups' Archimedes Newsletter. Many thanks to the editor, Tony Krzyzewski. Write to him if you want more details of Eureka c/o Barsons Computers, P O Box 26287, Epsom, Auckland, New Zealand.)

• **ROMs that work under the emulator.** Acornsoft's Comal, Prolog and Lisp seem at first look to be OK, and one reader comments that Logotron's Logo is OK but that the graphics are "funny" – whatever that means.

Damon Hoggett reckons that to get the View series ROM's to work, you need to *LOAD them at &10000 and then poke the following addresses to &EA (a NOP instruction) as explained last month. View B3.0: &128A2, ViewSheet B1.0: &10690, 691 and 692 and ViewStore 1.0: &12BCE, BCF and BD0 then you *SAVE filename 10000+4000 8000 8000. We made a mistake with the mention last month of Viewstore 1.1 (page 23). The poke should be ?&12BE2=&EA, not 1ABE2 and you should also poke the next two bytes, &12BE3 and E4.

BBCSoft's Monitor ROM seems to work OK with the emulator. You can apparently assemble, disassemble and single-step through 6502 machine code without problems.

Some folk are saying the Inter-Chart works under the emulator, though not option 9 to import data.

• **BASIC editor on 0.20 OS** – If you are in BASIC and you type EDIT, BASIC issues a *ARMBE command, so if that module is not already loaded into memory, it will look in the operating system ROM and then in the current directory on the current drive for the ARMBE module. If it is not there, it will come up with "Bad command". However, if it finds it, it will load it into memory. Then to enter the editor, you just type EDIT again. So, until you get your 1.2 OS(!) it is a good idea to copy the ARMBE module into the directory in which you keep your BASIC programs so that it is ready to use at any time.

• **No room in RMA** – If you are in BASIC and try to *RMLOAD a module, you may get "No room in RMA" even if the configuration is set to allow enough space for that particular module. But if you QUIT first into the Arthur supervisor you can do the RMLOAD and then go back into BASIC and OLD to get your program back. However, I think I would tend to save the program first just in case!

• **BASIC V tips from Colin Dean**, author of 'Advanced BASIC' (Tubelink's BASIC V look-alike for the BBC & Master)

In the LIST IF command, if you put a space between the IF and the <string> that follows it, you get a different effect. For example if you have two lines:

```
10RECTANGLE 1,2,3,4
20 RECTANGLE 5,6,7,8
```

Then `"LIST IF RECTANGLE"` shows line 20 only, whereas `"LIST IFRECTANGLE"` shows both.

A neat way to test more than one expression at once, without having to use heavily nested IF's is to use "CASE TRUE OF". For example,

```
CASE TRUE OF
  WHEN X=3 AND Y=4:PRINT"X=3 and Y=4"
  WHEN X>7, Z=0:PRINT"X>7 OR Z=0"
ENDCASE
```

However, you should beware of mixing numerics and logicals in CASE expressions. For example,

```
X = 6
CASE X OF
  WHEN TRUE : PRINT "TRUE"
  WHEN FALSE : PRINT "FALSE"
  OTHERWISE PRINT "SPURIOUS"
ENDCASE
```

this prints "SPURIOUS" because '6' is neither true (–1) or false (0).

• **Improved boot file for WWPlus** – The !BOOT file on the Archimedes Wordwise Plus discs is a BASIC program which checks whether the 6502 emulator is installed and if not loads it off the Welcome disc. If instead you copy 65arthur onto your Wordwise Plus discs you can use a simpler boot file which just says:

```
QUIT
65ARTHUR
WW+
```

and then do a *OPT 4 3 so that the computer EXEC's the boot file instead of running it. If you are using the disc version of CP-ROM, just add

```
:SELECT SEGMENT 8
:LOAD TEXT "$.CP-ROM.CPstart"
*FX138,0,152
```

to the boot file and it will start up the CP-ROM as well.

• **Have you seen the whale?!** – When playing Zarch, one or two folk have discovered a whale that appears in the sea (or is it a shark or a sea-monster?). You get 1000 points if you exterminate it and sometimes it "beaches" itself and becomes a much easier target. By the way, how are the scores going? I've just about managed to avoid being called a wet lettuce or a stuffed aubergine, but one reader, Malcolm Roberts says his son has reached 53,291.

(STOP PRESS! I've just seen it too – very fleetingly. It was bluish with a zig-zag fin on its back!) A

For Hints and Tips about using the ADFS, see page 44.

Questions and Answers

When writing in with queries, or even when writing to give us your hints and tips, it would be helpful if you could tell us which operating system you are using because often, something which works on one operating system won't work on another and vice versa.

Q: There's no fan in my 310. Have I been 'done'?

A: No, the fan is only needed when you start putting podules in, so they supply you with the fan when you buy the back-plane.

Q: The A305 and A310 have space for 1 Mbyte of ram on the main p.c.b. but can this be expanded. Similarly, can the A400 series be expanded beyond 4 Mbyte?

A: The 300 series machines have a memory controller chip (MEMC) which can access up to 4 Mb but the operating system will not, at present, support it. Hopefully that will change with the 1.2 operating system. The 400 series will take 4 Mb on the main board but it will, I gather, have the same MEMC as the 300 series and therefore will not be capable of simple expansion beyond 4Mb of ram. i.e. further expansion would require a new MEMC and a new operating system as well as the extra ram.

Q: How do I access the User port on the I/O podule?

A: I haven't tried it, because I haven't got the podule yet, but presumably Acorn will have made it compatible with the BBC micro by providing OSBYTEs 150 and 151 for reading and writing the user port, printer port and 1MHz bus. (Can anyone confirm this and send us a few handy hints?)

Q: The 300 series can only have two podules fitted. Will it be possible to increase that?

A: I very much doubt whether Acorn will be providing any upgrade path but perhaps third parties will do so. Any ideas anyone?

(There was a mistake in last month's Q & A. On the 400 series, the Winchester controller is on the main board, not on a podule, so you will have the full four podule slots available as standard.)

Q: Will the hard disc controller podule on the 300 series take up one of the two podule slots?

A: On the 400 series, the Winchester controller is on the main board, and does not require a separate podule, but on the 300 series it comes as a podule leaving you with only one slot left.

Q: When will the MS-DOS podule be available, how much will it cost and how fast will it run?

A: Three good questions! The answers are: Don't know, don't know and don't know! All I can do is make semi-informed guesses. Firstly, I doubt if it will be available from Acorn before mid '88, but it wouldn't surprise me if a third party took it up and beat them to it. (Not for the first time!) As for speed, you would expect it to run at the same speed as any equivalent PC compatible, but probably a bit faster as you have the ARM processor acting as a parallel I/O processor. (See also "Comments" on page 3.)

Q: Is the hard disk as supplied by Acorn, a special version for the Archimedes, or can any hard disk be connected to the controller podule?

A: I haven't received my hard disc yet (they are "due in at the end of November", Acorn Sales people say) but I imagine that it will be a standard hard drive and that any other drive would do to replace it. The problem is that Acorn will not supply the podule on its own! It's one purchase – "podule-plus-hard-drive" at £499.

Q: What exactly is offered by your technical help service? Can you give an example of when it might be of help?

A: The THS allows you to ring us up during normal office hours (8.30 – 5.30, Mon – Fri + some Saturday a.m.'s) with urgent technical queries. If we cannot answer the question immediately, we will endeavour to find an answer from one of our many contacts in the Archimedes world (both inside and outside Acorn) and will ring you back as soon as possible. For example, the gentleman who discovered the "Naff RTC month" error (page 9 last month) was able, on the day the problem occurred, to ring us up a couple of times as we tried to sort out his problems. The alternative would have been a lengthy correspondence with us or with Acorn's technical support team.

Q: How do I put notes in the diary for 1988?

A: The only way I have found is to *SET SYS$YEAR 1988 before running the desktop and then remember to *SET SYS$YEAR 1987 afterwards.

Q: In an earlier Archive you mentioned "using <ctrl-L> to clear the screen". What do the bits inside the triangular brackets mean?

A: Sorry about that! I had got so used to using that convention that I forgot to explain it. Firstly anything between triangular brackets refers to a keystroke of some sort, so "use <escape> to do whatever" means that you press the escape key. Then you get things like <shift-break> which means hold down the shift key and press and release the break key, so <ctrl-L> means hold the Ctrl key and press and release the L key.

Help!!!??

• Mr J M Smith would be interested to hear from anyone doing any programs to do with stocks & shares? Let us know if you are doing anything.

• One of our readers is having problems when using the assembler with using EQUS FNfunctionname (parameters). Small programs seem to be OK but when using it with large programs it crashes during assembly with unpredictable results… sometimes hanging, sometimes starting up the drive and occasionally failing with a memory exception error. It is not a problem with violations of the assembled code because doing the same job by coming out of the assembler and using a PROC, works OK. Anyone else with similar problems?

• Another reader finds that on some occasions, especially after trying a program that won't run, the BREAK key acts as it did on the BBC, i.e. clears the screen and puts up the usual message. Anyone with similar experience?

• Has anyone had problems with ARM machine code programs that run when you assemble them from BASIC but not when you try to *RUN them later? One reader got error messages about "links" when he tried this.

• Does anyone know of any programs available to do with heraldry, asks Malcolm Roberts, or is anyone working on anything?

How fast is Archimedes?

Ronald Alpiar

Just how much faster is the Archimedes than other machines? – There is no simple answer to this question, since it all depends both on what you want to do and how you choose to do it.

This article will try to shed some new light by concentrating on two different algorithms – the 'Savage Benchmark' and the 'Ackerman Function'. They are both practically independent of I/O devices, and hence their performance tests pure internal computation speed and efficiency, but at that point, all similarity between the two tests ends.

The Savage Benchmark – this is essentially a test for speed and accuracy in performing the floating point arithmetic of elementary mathematical functions. First described in "Byte" (Vol 10, no.11, page 67, 1985), it essentially consists of a 2499-fold loop of direct and inverse functions, symbolically defined as:

```
A=1
do 2499 times over:
A=tan(arctan(exp(loge(sqr(A*A)))))+1
```

Clearly the theoretical result should be A=2500.

It was tested out on a number of computers, and in three languages – BASIC, FORTRAN 77 (both single & double precision) and ANSI 'C'.

The speed of Archimedes BASIC when running the Savage benchmark is truly staggering – all the more so, when one considers that, as an interpretive language, it appears to be twice as fast as the compiled languages, FORTRAN and

Machine	Language	Error	Time (Secs)
BBC Master	BASIC	–3.14E–1	140.05
BBC Master + Turbo	BASIC	–3.14E–1	68.26
BBC Master + 512K (80186 10 Mhz)	ANSI C	+8.41E–7	121.08
Cambridge Workstation, no fpu	BASIC	–3.14E–1	33.07
Cambridge Workstation, with fpu	BASIC	–3.14E–1	5.99
Cambridge Workstation, with fpu	ANSI C	+3.17E–7	8.23
Cambridge Workstation, with fpu	FORTRAN 77 single	–3.67E+2	5.66
Cambridge Workstation, with fpu	FORTRAN 77 double	–1.26E–6	7.92
IBM clone 8086 at 8 Mhz	Turbo Pascal	+4.63E–3	59
IBM clone with 8087 maths fpu	Turbo Pascal	+1.18E–9	6
IBM clone 80286	Turbo Pascal	+4.63E–3	54
IBM clone with 80287 maths fpu	Turbo Pascal	+1.18E–9	6
CRAY X-MP (without vectoring)	FORTRAN 1.14	–4.80E–20	0.746
Archimedes A310	BASIC V	–1.50E+0	4.52 3,93
Archimedes A310	RAM-BASIC V	–1.50E+0	3.08 2,84
Archimedes A310	FORTRAN 77 single	–2.28E+1	8.68
Archimedes A310	FORTRAN 77 double	+1.18E–9	8.96
Archimedes A310	ANSI C	+1.18E–9	9.42

Figure 1: Time taken and accuracy when performing the Savage benchmark

C. One begins to wonder whether it's worth the hassle of programming in a compiled language, unless high accuracy is essential.

Notice that the addition of floating point hardware (fpu) speeds up the calculation by a factor of 5 to 10. It will be very interesting to see what acceleration the forthcoming ARM fpu will offer. If the factor of 5 to 10 is anything to go by, this would make Archimedes' BASIC speed comparable to the fastest multi-million dollar computer in existence!

Comparison of accuracies is also surprising. With its 32-bit word length, why is the Archimedes no more accurate than the 8-bit Master? The answer lies in how BASIC represents floating point numbers – only 5 bytes (40 bits) in each case – and in how roundoff errors are handled. To gain higher accuracy, a more faithful representation of floating point quantities, using more bytes, is needed.

The Ackerman function, ACK(M,N), is an integer function of the two integer parameters M and N. Its definition is essentially recursive. That is, there's no way to define its value for given parameters, without reference to its value for smaller parameters. Here's the definition:

```
ACK(0,N)  = N + 1
ACK(M,0)  = ACK(M-1,1)
ACK(M,N)  = ACK(M-1, ACK(M,N-1))
```

This function was first formulated by the mathematical-logician W.Ackerman in 'Zum Hilbertschen Aufbau der reel Zahlen' (trans.'On the Construction of Real Numbers by the Method of Hilbert') in Math.Ann.,99 (1928) pp.118-133. Originally proposed as a function of three parameters, the two parameter simplification above is the one usually quoted. Ackerman proved that the function grows faster than any definable 'primitive recursive function' – which means roughly speaking any function you or I could construct using exponents, factorials etc.etc. Its importance lies in the theory of sets, in propositional calculus and the logical foundations of mathematics. It also has connections with Godel's proof of the undecidability of certain propositions within any formal mathematical system (the so-called incompleteness theorem). In other words it's a pretty powerful animal!

For small values of the parameters (M<4, N<10 or so) the function is fairly easy to compute. But by the time M=4 it really takes off. ACK(4,0) is likewise reasonably amenable. However I estimate that it would take Archimedes' BASIC two solid **weeks** to perform ACK(4,1)! As for ACK(4,2), one can confidently claim that no machine, be it ever so fast, or run ever so long, could possibly complete the algorithm.

If that seems a rash claim, consider this. The time taken to compute the algorithm is roughly proportional to the number of recursive steps needed. A reasonable esimate for the number of recursions to calculate ACK(4,2) is 10E4390 (10 to the power 4390). Now machine code ARM can be timed and calculates at about 10E11 recursions/year. This means that it would take 10E4379 years to calculate ACK(4,2). Even if computer speeds were increased by an unimaginable billion-billion (10E24) times, it would still take 10E4355 years. This time span may be compared with the paltry present age of the universe since the Big Bang (10E10 years), or even the Grand Unification Theories' (GUTs) estimate of the total life of the universe, based upon proton decay time of a mere 10E31 years. Of course, long before then, all the available space in the universe for storage of intermediate results (10E40 electrons and protons) would have been exhausted!

Notwithstanding all that, one can write down a simple arithmetic expression of 9 characters, maybe less, for the value of ACK(4,2). In this context, characters are drawn from the digits 0-

9,),(, +,-,*, / and the exponent symbol ^. No maths beyond O levels is needed. The first ACK(4,0) readers to submit correct solutions will be rewarded by seeing their names acknowledged in print!

For testing this function, FORTRAN is ruled out since it doesn't allow subroutines to be recursive. The algorithm was therefore tested in BASIC (listing L5), in ANSI 'C' (listing L6) and in Archimedes ARM code (listing L7).

The algorithm for the Ackerman function challenges not only the speed, but also the stamina of the software. The first refusals to tackle the ACK(3,n) hurdle are indicated by stars in the tabulation, the reason usually being stack overflow. Some compilers (e.g. Borland's Turbo C) let the user specify stack sizes larger than the default value. In the latter case I upped stack size to 40000 bytes to compute ACK(3,8). Other compilers are less accommodating. Beebug's C, apparently allocates only 256 bytes to the 'function stack'; hence it expires with a 'Too many nested functions' error message as early as ACK(3,4). Acorn's ANSI C compiler on the Archimedes is a pre-release version, likewise with a fixed, non-expandable stack of 16k bytes – it gives up on reaching ACK(3,6). However Archimedes BASIC seems to know no such limit. It easily calculated ACK(3,9), and may have sailed on to reach ACK(3,10) but for the author's reluctance to run the machne for an estimated 5 hours. The same applies to the ARM machine code version of the algorithm, which actually completed ACK(3,12) in 3008 seconds, and might well, had I permitted, have soldiered bravely on to ACK(3,13)!

As far as speed is concerned, this algorithm certainly separates the men from the boys. Contrast BASIC's brilliant behaviour with Savage, with its pedestrian and plodding performance here. Why so? Simple expanation: in Savage, BASIC had actually very little interpretation work to do – beyond counting a simple loop and popping in and out of machine code routines for the maths functions – where all the real number crunching was done. Little wonder that it compared favourably with the compilers. However the Ackerman Function offers BASIC no such opportunities for passing the buck; it's forced to do its own dirty work. Nevetheless, in its favour are its stamina and 10-fold speed improvement over the Master.

Machine	Language	N=2	N=3	N=4	N=5	N=6	N=7	N=8	N=9
BBC Master	BASIC	2.2	9.8	41	171	✻			
BBC Master + turbo	BASIC	1.1	4.8	20	83	✻			
	BEEBUG C	1.4	6.2	✻					
BBC Master + 512K	Borland C	0.02	0.08	0.36	1.8	6.1	24	98	✻
	Zorland C	0.62	2.5	10	✻				
	Borland TB	0.27	0.77	2.7	10	42	✻		
Cambridge Workst'n	BASIC	3.0	12	51	210	✻			
	ANSI C	0.03	0.09	0.38	1.5	6.1	25	99	✻
Archimedes A310	BASIC V	0.23	1.0	4.4	18	74	299	1200	4807
	RAM-BASIC	0.19	0.83	3.5	15	59	237	950	3807
	ANSI C	<0.01	0.02	0.08	0.33	1.45	✻		
	ARM		0.01	0.04	0.18	0.72	2.9	12	47

Figure 2: Time taken (secs) to perform ACK(3,N) (✻ the compiler gave up at that stage)

Without the slightest doubt, the real star of the show is the ARM machine code version of the algorithm, beating BASIC by a factor of 100!

Acknowledgements

The author wishes to acknowledge: Department of Physics and Astronomy, University of Glasgow for results relating to the Cambridge Workstation and Master Turbo. Also The Sky Publishing Corporation, USA, for the results relating to the IBM and Cray machines.

L1: The Savage Benchmark in BASIC

```
10 A=1 : TIME=0
20 FOR I%=1 TO 2499
30    A=TAN(ATN(EXP(LN(SQR(A*A)))))+1
40    NEXT
50 T=TIME/100
60 PRINT" Answer=";A;" time=";T;"
seconds"
70 END
```

L2:The Savage Benchmark in FORTRAN (single precision)

```
      PROGRAM SAVAGE
C     NO ACCESS TO INTERVAL TIMER
C     IN ACORN ARCHIMEDES FORTRAN 77
      INTEGER I
      REAL A
      A=1.
      DO 1 I=1,2499
1     A=TAN(ATN(EXP(LOG(SQRT(A*A)))))+1.
      PRINT 10,A
10    FORMAT ( ' ', E22,17)
      END
```

L3: The Savage Benchmark in FORTRAN (double precision)

```
  PROGRAM SAVAGED
  INTEGER I
  DOUBLE PRECISION A
  REAL A
  A=1.
  DO 1 I=1,2499
1 A=DTAN(DATN(DEXP(DLOG(DSQRT(A*A)))))+1.
  PRINT 10,A
10 FORMAT ( ' ', E22,17)
  END
```

L4: The Savage Algorithm in 'C'

```
#include <stdio.h>
#include <math.h>
#include <time.h>
void main ()
{
  int i;
  long tix,tiy;
  double a=1.0;
  tix=clock();
  for (i=1; i<2500, i++)
    a=tan(atan(exp(log(sqrt(a*a)))))+1;
  tiy=clock();
  printf("\n savage=%30.20e",a);
  printf("\n time taken=%ld centi-
seconds",tiy-tix);
}
```

L5: The Ackerman Function in BASIC

```
 10 INPUT "M,N?",M%,N%
 20 TIME=0
 30 Z%=FNACK(M%,N%)
 40 A=TIME/100
 50 PRINT"Time=";A;"seconds,ACK=";Z%
 60 END
100 DEF FNACK(M%,N%)
110 IF M%=0 =N%+1
120 IF N%=0 =FNACK(M%-1,1)
130 =FNACK(M%-1,FNACK(M%,N%-1))
```

L6: The Ackerman Function in 'C'

```
#include <stdio.h>
#include <time.h>
void main ()
{
  int a,b,ans;
  long tix,tiy;
  puts("\n Key in 1st parameter");
  scanf("%d",&a);
  puts("\n Key in 2nd parameter");
  scanf("%d",&b);
  tix=clock();
  ans=ackerman(a,b);
  tiy=clock();
  printf("\n Ackerman(%d , %d),=%d",
                              a,b,ans);
  printf("\n time taken=%ld
centiseconds", tiy-tix);
}
```

```
 int ackerman (m,n)
 int m,n
{
 if (m==0) return (n+1);
 else if (n==0) return ackerman(m-1,1);
 else return ackerman(m-1,ackerman(m,n-1));
}
```

L7: The Ackerman Function in ARM machine code

```
 10 DIM org 300                 reserve space for code
 20 link=14                     return address in register R14
 30 sp=13                       stack pointer in register R13
 40 FOR pass=0 TO 2 STEP 2      usual double pass assembly
 50 P%=org                      start address
 60 [ OPT pass                  set OPT according to which pass
 70 .ack                        label of start of algorithm on entry M in R1,
                                N in R2 on exit answer in R0
 80 CPM R1,#0                   test for M=0
 90 BNE mgt                     if not goto 'mgt'
100 ADD R0,R2,#1                case M=0, answer=N+1
110 MOV PC,link                 return, setting link address in PC
120 .mgt                        case M >0
130 CMP R2,#0                   test for N=0
140 BNE both                    if not goto 'both'
150 STMFD (sp)!,{R1-R2,PC}      save M,N & link address in stack
160 SUB R1,R1,#1                M=M-1
170 MOV R2,#1                   N=1
180 BL ack                      recursively re-enter Ackerman
190 LDMFD (sp)!,{R1-R2,link} restore M, N & link address
200 .both                       case M,N both >0
210 SDMFD (sp)!,{R1-R2,link} as l.150
220 SUB R2,R2,#1                N=N-1
230 BL ack                      as l.180, get ACK(M,N-1) into R0
240 LDMFD (sp)!,{R1-R2,lnk}     as l.190
250 MOV R2,R0                   N=ACK(M,N-1)
260 SUB R1,R1,#1                M=M-1
270 STMFD (sp)!,{R1-R2,link} as l.150
280 BL ack                      as l.180, get answer in R0
290 LDMFD (sp)!,{R1-R2,link} as l.190
300 ]
310 NEXT                        next pass of assembler
320 INPUT"M,N?",B%,C%           put M in R1 and N in R2
330 TIME=0
340 PRINT USR ack               jump into machine code, print answer
350 PRINT "Time=";TIME/100;" secs"
360 END
```

Archimedes Colour Blender

Ian Nicholls

The purpose of the program is to enable you to mix a set of 6 colours from the 4096 colours available on the Archimedes. You can then use them in your own programs, or to modify other peoples programs to suit your own taste!

The diagram below shows the screen layout for the display produced by the program. The 6 colours are displayed as if you were designing a text-based program, with up to three simultaneous colours for the text, a background colour, a coloured border round the edge of the screen, and a rectangular frame between the border and the background. This is the "main panel" display.

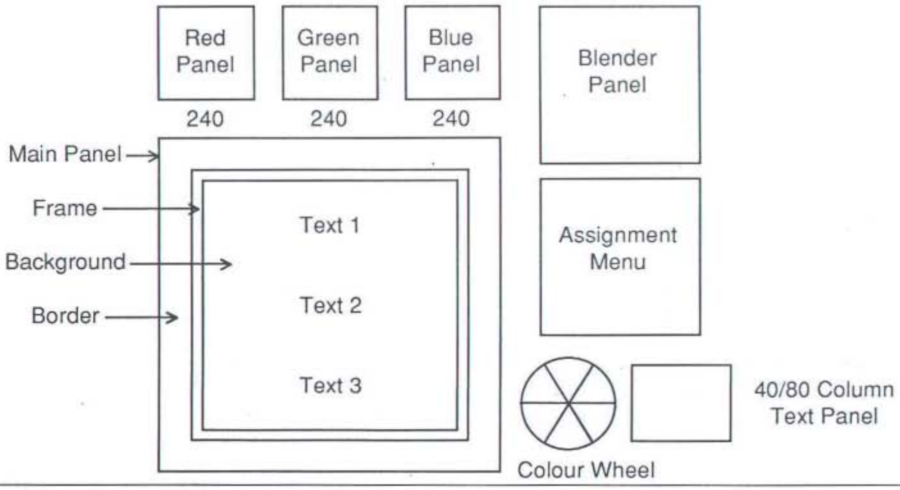
The 6 colours are also shown in the six sectors of the "colour wheel" at the bottom right of the screen. Since you may want to use a 40 column text mode instead of the 80 column one of the blender program, the three blocks of text in the "main panel" display can be switched between 80 and 40 columns.

All of the interaction with the program is via the mouse. The colour you are mixing is shown in the "blender panel" at the top right. If you move the mouse pointer onto this panel and press the middle mouse button, the colour of the panel changes from white, through progressively darker shades of grey, to black. To stop at any shade take your finger off the button. To move back towards white again, press the left-hand mouse button.

As you change the blender panel, the intensities of red, green and blue panels all go up or down in step. To alter one of these separately from the others (i.e. to obtain a colour other than white, grey or black in the blender panel), move the pointer onto the panel you wish to alter and use the buttons as above.

To use the blended colour in your program, you need to know what the intensities of red, green and blue are that go to make it up; if red, green and blue respectively were 240, 128 and 48: to make colour "col%" in your program the same as the blended shade, you would use the statement:-

```
COLOUR col%, 240,128,48
```



(N.B. If you wanted to make the standard Archimedes border into this shade you would have to use the statement, VDU 19,0,24,240, 128,48 see page 371 of User Guide issue 1.)

So far all we have done is to mix a colour: we need to be able to make the text, or the background, etc in the "main panel" the same as this colour. To do this move the pointer onto the "assignment menu". This menu consists of the following items:-

Text 1
Text 2
Text 3
Background
Frame
Border

with the numbers after each, which are the intensities of the red, green and blue components of their current colour.

To make, say, the border the same as the colour in the blender panel, move the pointer onto the word "Border" in the menu and press the left-hand mouse button. Tthe word is then highlighted and the border in the main panel changes to its new colour. While the word "Border" is highlighted, the border colour will change as you change the colour of the blender panel.

When you are happy with the colour of the border, move the pointer back onto the word "Border", press the left-hand mouse button and the highlighting will be removed. You can now alter the blender panel colour without also altering the border colour. The same process can be used to alter the colour of any other of the items in the "assignment menu".

The last feature of the program is to allow you to swap between 80 and 40 columns for the three blocks of text in the main panel. To begin with, the text is printed in 80 columns, and the bottom right-hand box says "TEXT 80 COL". The "80" changes to a "40" and the text in the main panel is re-written in the same colours but in 40 columns. The printing is much slower than normal because a large amount of calculation is taking place in PROCOsword. This procedure takes a letter, looks up the pattern of eight bytes from which it is made (using OSWORD 10) and defines two new characters (ASCII codes 130 and 131) which will print the original character, but twice as wide as before.

To return to 80 column text, move the pointer into the "TEXT 40 COL" box and press the left-hand mouse button. To leave the program press <escape>.

For those of you interested in how the program works, most of it is reasonable straight-forward. The major feature is the use of the "SYS" BASIC keyword, to call the operating system routine "OSWORD". OSWORD 10 takes the character whose ASCII code has been placed in location "char" and returns its eight byte definition in locations char+1 to char+8. If you want to unravel the rest of PROCosword, it will help you to write out a% (256) and b% (254) as binary numbers!

The other parts of the program worthy of comment are the statements like `red%-=red%<15` in lines 530 to 600. They are used to increase or decrease the red, green and blue intensities in the panels at the top of the screen, which may take integer values between 0 and 5. The statement is exactly equivalent to :-

```
red%=red%-(red%<15)
```

Now if red% is less than 15, then (red%<15) is true and returns the value -1, but if red%=15 then (red%<15) is false and returns the value 0. So the statement is exactly equivalent to :-

```
If red%<15 THEN red%=red%+1 ELSE
red%=red%
```

```
 10 REM > BLENDER
 20 REM   Archive Colour Blender
 30 REM   Ian Nicholls, 5 November 1987
 40 :
 50 REM   Initialisation
 60 :
 70 ONERRORPROCerror:END
 80 DIM flag%(6),A$(6),bit%(7),left_char%(8),right_char%(8),x%(6),y%(6)
 90 DIM char 8
100 MODE12:OFF:panel%=999:col_flag%=1:red%=15:green%=15:blue%=15
110 RESTORE:FORI%=1TO13:READc%,r%,g%,b%:COLOURc%,r%,g%,b%:NEXT
120 FORI%=1TO6:READA$(I%):NEXT
130 x=72*COS(PI/3):y=72*SIN(PI/3):x%(0)=72:x%(1)=x:x%(2)=-x:x%(3)=-72
140 x%(4)=-x:x%(5)=x:x%(6)=72:y%(1)=y:y%(2)=y:y%(4)=-y:y%(5)=-y
150 COLOUR15:COLOUR128:PROCprint_40(17,0,"ARCHIVE COLOUR BLENDER",0)
160 FORI%=1TO6:flag%(I%)=-1:NEXT
170 FORI%=1TO3
180   A%=40+248*(I%-1)
190   GCOL0,15:MOVEA%,984:MOVEA%,768:PLOT85,A%+216,984
200   GCOL0,13:PLOT85,A%+216,768
210   GCOL0,I%:RECTANGLE FILL A%+16,784,184,184
220 NEXT
230 GCOL0,15:MOVE804,986:MOVE804,568:PLOT85,1220,986
240 GCOL0,13:PLOT85,1220,568
250 GCOL0,4:RECTANGLE FILL 820,584,384,384
260 COLOUR15:PRINTTAB(65,16)"0,0,80"TAB(65,18)"0,80,16"TAB(65,20)"112,
     0,64"TAB(65,22)"208,208,128"TAB(65,24)"48,16,0"TAB(65,26)"112,48,16"
270 PROCtext:PROCpanel:PROCgraphics:*POINTER 1
280 MOUSE ON:MOUSE TO 640,512
290 :
300 REPEAT
310   panel%=999:FORI%=1TO6:IFflag%(I%)=1  panel%=I%+4
320   NEXT
330   IFpanel%<>999 COLOURpanel%,red%*16,green%*16,blue%*16
340   TIME=0:REPEAT UNTIL TIME=17
350   PROCread_mouse
360 UNTIL0
370 :
380 DEFPROCread_mouse
390 MOUSE X%,Y%,button%
400 IFY%>583 AND Y%<969 PROCalter_colours
410 IFY%<512 AND Y%>159 AND button%=4 PROCalter_text
420 IFY%<141 AND Y%>15 AND X%>979 AND X%<1231 PROCcolumns
430 PROCtext
440 ENDPROC
450 :
```

```
460 DEFPROCalter_colours
470 colour%=999
480 IF X%>55 AND X%<241 AND Y%>783 colour%=1
490 IF X%>303 AND X%<491 AND Y%>783 colour%=2
500 IF X%>551 AND X%<737 AND Y%>783 colour%=3
510 IF X%>819 AND X%<1205 colour%=4
520 IFcolour%=999 ENDPROC
530 IFbutton%=4 AND colour%=1 red%-=red%<15
540 IFbutton%=2 AND colour%=1 red%+=red%>0
550 IFbutton%=4 AND colour%=2 green%-=green%<15
560 IFbutton%=2 AND colour%=2 green%+=green%>0
570 IFbutton%=4 AND colour%=3 blue%-=blue%<15
580 IFbutton%=2 AND colour%=3 blue%+=blue%>0
590 IFbutton%=4 AND colour%=4 red%-=red%<15:green%-=green%<15:blue%-=
                                                               blue%<15
600 IFbutton%=2 AND colour%=4 red%+=red%>0:green%+=green%>0:blue%+=
610 WAIT:COLOUR1,red%*16,0,0:COLOUR2,0,green%*16,0              blue%>0
620 COLOUR3,0,0,blue%*16:COLOUR4,red%*16,green%*16,blue%*16
630 ENDPROC
640 :
650 DEFPROCtext
660 FORI%=1TO6
670   IFflag%(I%)=1 THEN COLOUR143:COLOUR0 ELSE COLOUR128:COLOUR15
680   PRINTTAB(51,14+2*I%);A$(I%)
690   COLOUR128:COLOUR15:PRINTTAB(62,14+2*I%);"-"
700 NEXT
710 PRINTTAB(7,9);red%*16;"  "TAB(23,9);green%*16;"  "TAB(38,9);
720 IFpanel%=999 ENDPROC                                   blue%*16"  "
730 X$=STR$(red%*16):Y$=STR$(green%*16):Z$=STR$(blue%*16)
740 X$=X$+","+Y$+","+Z$:len%=LENX$:Y$=""
750 IFlen%<11 Y$=STRING$(11-len%," ")
760 X$=X$+Y$:PRINTTAB(65,6+2*panel%);X$;
770 ENDPROC
780 :
790 DEFPROCalter_text
800 IFX%<814 OR X%>975 ENDPROC
810 IFY%<192 AND X%<912 flag%(6)=-flag%(6):IFflag%(6)=1 PROCflag(6):E.
820 IFY%>223 AND Y%<256 AND X%<896 flag%(5)=-flag%(5):IFflag%(5)=1
                                                         PROCflag(5):E.
830 IFY%>287 AND Y%<320 flag%(4)=-flag%(4):IFflag%(4)=1 PROCflag(4):E.
840 IFX%>912 ENDPROC
850 IFY%>351 AND Y%<384 flag%(3)=-flag%(3):IFflag%(3) PROCflag(3):E.
860 IFY%>417 AND Y%<448 flag%(2)=-flag%(2):IFflag%(2) PROCflag(2):E.
870 IFY%>480 flag%(1)=-flag%(1):IFflag%(1) PROCflag(1):ENDPROC
880 ENDPROC
890 :
 900 DEFPROCcolumns
 910 IFbutton%=4 THEN col_flag%=-col_flag% ELSE ENDPROC
 920 COLOUR15:COLOUR128
 930 IF col_flag%=-1 THEN
 940   PROCprint_40(63,30,"4",0):PROCtext_40:*FX21,9
 950   ELSE PROCprint_40(63,30,"8",0):PROCtext_80
 960 ENDIF
 970 ENDPROC
 980 :
 990 DEFPROCpanel
1000 GCOL0,10:RECTANGLE FILL 40,0,712,636
1010 GCOL0,9:RECTANGLE FILL 72,32,648,572
1020 PROCtext_80
1030 GCOL0,15:MOVE980,16:DRAW980,140:DRAW1230,140:MOVE982,20:DRAW982,136
1040 GCOL0,13:MOVE1230,140:DRAW1230,16:DRAW980,16:MOVE1228,136:DRAW1228,20
1050 COLOUR15:COLOUR128
1060 PROCprint_40(65,28,"TEXT",0):PROCprint_40(63,30,"80 COL",0)
1070 ENDPROC
1080 :
1090 DEFPROCtext_80
1100 GCOL0,8:RECTANGLE FILL 80,40,632,556:COLOUR136
1110 FORI%=1TO3:COLOUR(I%+4)
1120   PRINTTAB(7,10+5*I%)"The quick brown fox jumps over the"TAB(7,
                                                   11+5*I%)"lazy dog!"
1130   PRINTTAB(7,12+5*I%)"1234567890-!@#$%^&*()_+$#{}[]|\<>?"
                       TAB(7,13+5*I%)"/,.:;~`  -=-=-=-  ACORN ARCHIMEDES"
1140 NEXT
1150 ENDPROC
1160 :
1170 DEFPROCgraphics
1180 ORIGIN 884,72
1190 FORI%=0TO5
1200   GCOL0,(I%+5):MOVE0,0:MOVEx%(I%),y%(I%):PLOT&B5,x%(I%+1),y%(I%+1)
1210 NEXT:ORIGIN0,0
1220 ENDPROC
1230 :
1240 DEFPROCtext_40
1250 GCOL0,8:RECTANGLE FILL 80,40,632,556:COLOUR136
1260 PROCprint_40(5,15,"The  quick  brown",1)
1270 PROCprint_40(5,16,"fox jumps over the",1)
1280 PROCprint_40(5,17,"lazy  dog!",1)
1290 PROCprint_40(5,18,"1234567890 @#$%&*$",1)
1300 ENDPROC
1310 :
```

```
1320 DEFPROCprint_40(L%,M%,A$,p%)
1330 LOCAL I%,J%
1340 IFp%=0 PRINTTAB(L%,M%);
1350 FORI%=1TOLEN(A$)
1360   B$=MID$(A$,I%,1):PROCosword(B$):IFp%=0 THEN
1370     VDU130,131
1380     ELSE FORJ%=0TO2
1390       COLOUR(5+J%):PRINTTAB(L%+I%*2,M%+5*J%);CHR$130;CHR$131;
1400     NEXT
1410   ENDIF
1420 NEXT
1430 ENDPROC
1450 DEFPROCosword(B$)
1460 LOCAL I%,J%
1470 ?char=ASC(B$):SYS "OS_Word",10,char TO ,char
1480 FORJ%=1TO8
1490   a%=256:b%=254:Z%=char?J%
1500   FORI%=7TO0STEP-1
1510     a%=a%>>1:b%=b%>>1
1520     bit%(I%)=(Z% AND a%) DIV a%:Z%=Z% AND b%
1530   NEXT
1540   left%=0:right%=0:count%=1
1550   FORI%=4TO7
1560     left%=left%+bit%(I%)*count%:left%=left%+bit%(I%)*count%*2
1570     right%=right%+bit%(I%-4)*count%:right%=right%+bit%(I%-4)*count%*2
1580     count%=count%*4
1590   NEXT
1600   left_char%(J%)=left%:right_char%(J%)=right%
1610 NEXT
1620 VDU23,130:FORJ%=1TO8:VDUleft_char%(J%):NEXT
1630 VDU23,131:FORJ%=1TO8:VDUright_char%(J%):NEXT
1640 ENDPROC
1650 :
1660 DEFPROCflag(J%)
1670 FORK%=1TO6
1680   IFK%<>J% flag%(K%)=-1
1690 NEXT
1700 ENDPROC
1710 :
1720 DEFPROCerror
1730 VDU2:ON:CLS:MOUSE OFF:IF ERR=17 ENDPROC
1740 REPORT:PRINT" at line ";ERL
1750 ENDPROC
1760 :
1770 DATA 0,0,0,0,1,240,0,0,2,0,240,0,3,0,0,240,4,240,240,240,5,0,0,80
1780 DATA 6,0,80,16,7,112,0,64,8,208,208,128,9,48,16,0,10,112,48,16,13
1790 DATA 128,128,128,15,240,240,240
1800 DATA "Text 1","Text 2","Text 3","Background","Frame","Border"
```

BBC/Archimedes File Transfer Kit

A review by Mark Seeley

Archive Vol 1 No 2 page 3 mentioned a kit for transferring software from 5.25" discs via the RS423 interfaces to Archimedes 3.5" format using a BBC B effectively as a terminal. We have now had an opportunity to try out this product and can report very favourably.

You get a ready-made lead with plugs at each end and clear instructions to insert the BBC DIN plug with the gap on the metal shield uppermost, which is easy not to do correctly. Then there are two discs, one for each computer. Boot them up, BBC first, and click on the appropriate icons in Brainsoft's enhanced Desktop.

The disc drive (DFS only for the moment, but there are plans for an ADFS version soon) on the BBC whirs. Select the file you want from the distant (5.25") disc, select Load and it is transferred to your Archimedes drive 0 automatically. Should it be of the wrong type to run straight away, there are helpful hints about the *SETTYPE Operating System command in the rather temporary-looking manual.

The advantages of the package are its relative cheapness, the fact that it works reliably and that baud rates can be changed if files should ever be corrupted in the process. Use of the WIMP environment makes it easy to use.

Although I found supporting documentation rather meagre, those who have files to transfer from the older format are likely to have experience of such things and an understanding that it is in the nature of communications, say, to have to re-run the main file on the BBC after performing certain checking operations. As the blurb says, It is "a low-cost quick method".

The major disadvantage is in tying up a BBC computer instead of just using another disc-drive, though the latter method still seems fraught with snags – no reliable interface kit is available at the time of writing and in any case it will be harder still if you upgrade your Archimedes to have two internal disc-drives.

Many programs of true simplicity are appearing "to get you started" and in these early days when all that is needed here is to get the files across to the Archimedes once and for all, this kit from Brainsoft can be thoroughly recommended.

The transfer kit costs just £14 and is available from Brainsoft, 22 Baker Street, London, W1M 1DF.

Load of Rubbish

I sent a copy of Archive to John Lewis, the editor of Mac User UK, to see if he could give me any constructive criticism of the magazine in terms of layout. Needless to say, his magazine is done on an Apple Mac and Laser-writer just as this one is, so I thought he might be helpful.

He made one or two helpful comments, and in particular said that the gutter between the columns is a bit narrow, but I've left it the same width so that I can get in as much text as I possibly can.

He also said that he didn't like the blank spaces at the bottoms of pages. He suggested I spread out the text a bit. I tried it in the article above but didn't like the effect. I have more material I could have put in, but nothing short enough to fit, so instead I decided to fill the space with this! Well, there's no blank space, is there?!

Using the WIMP Environment – Part 2

Adrian Look

In last month's article I explained how to set up windows in the WIMP enviroment. The subject of this second article is how BASIC interacts with the Arthur Window Manager (AWM). The AWM uses a lot of 'data blocks' and, in order to make sense of this article, you should become familiar with that idea first.

A 'data block' is an area in memory which stores information to be used by specific routines. The data is stored, not like BASIC variables, but as raw integer numbers. The relevant information can be located in the data block by means of an 'offset', i.e. a number which tells us where the data is located in terms of the number of bytes from the start of the block. A routine's offsets are predetermined and so any data block should comply with the routine's format. If it doesn't, the routine will not work properly.

A data block is recognised by the routine by its position in the memory which is referred to as the address of the data block. Memory may be set aside for these data blocks by the DIM statement and the information may be stored or retrieved via the indirection operators (! and ?). Often this data is stored as 'flags' (an application of the binary system) in order to save space.

Both these last two ideas are well documented in the User Guide, so I recommend that you look up binary/hex, indirection operators and the DIM statement. (If you only have the provisional User Guide that has no index, you could of course use Matthew Treagus' index which he produced for us in last month's magazine!)

'Wimp_Poll' at &400C7

This routine is the heart of the WIMP enviroment. It is the interface between the AWM and the application program. On entry, this routine requires the address of a data block, (in which it returns to the application different information depending on what has happened in the WIMP enviroment) and a 'mask'.

This routine allows the AWM to update itself (perform any processing nessesary) and check what is happening (i.e. what the user is doing), then it reports back to the application. The 'mask' tells the AWM which conditions should NOT be reported to the application. The conditions which may be masked out are:

- null reason code
- redraw window request
- pointer leaving window
- pointer entering window
- mouse click
- key pressed

The routine returns to the application with a 'reason code'. This 'reason code' tells the application what has happened and how to react, if at all. The full list of what can occur is shown below:

Reason conditions:	**Code:**
null reason code	0
redraw window request	1
open window request	2
close window request	3
pointer leaving window	4
pointer entering window	5
mouse clicked	6
user drag box	7
key pressed	8
menu select	9
scroll request	10

'Null reason code' – This code tells the application that it is free to go off and do its own processing because nothing has happened.

'Open window request' – This code tells the application that either a new window is to be opened or a presently displayed window is being changed. e.g. size, dragging, or scrolling. The application should call the 'Open_Window' routine (described last month) on receipt of this condition. However, the 'Open_Window' procedure does not need to set up a data block because the 'Wimp_Poll' routine has already done this for the application in its own data block.

'Close window request' – When the user clicks the 'quit' box of a window its handle is returned along with this 'reason' code. The handle is stored in the 'Wimp_Poll' data block so, as in the 'Open _Window' condition, the AWM routine can be called straight away ('Close_Window'). If for some reason the window should not be closed at that point, the application does not have to close it. It could, for example, open an error box saying that the window should not be closed. Similarly, if one window is closed the application may wish to close another window as well. Obviously the options are entirely up to you, the programmer!

'Pointer entering window' and **'pointer leaving window'** – These routines are used to notify the application when the pointer is moved in or out of a window. This information should be used to set the pointer shape (this will be discussed in a later article). The pointer shape should only be set when in a window and should be reset when it has left the window. Both conditions return the handle of the window being entered or left in the 'Wimp_Poll' data block.

'User drag box' – This code notifies the application that the 'drag' operation has finished. A 'drag' operation is any operation that causes the window to move in any way e.g. size change, 'picking the window up', etc. It is just another way of keeping track of what is going on in the WIMP enviroment

'Scroll request' – This code notifies the application that the window area should be scrolled. It allows the user to alter the amount by which the area is scrolled. The 'Wimp_Poll' routine returns the following in its data block:

0 – window handle
4 – work area coordinates (x0,y0,x1,y1)
20 – x,y scroll positions (scx,scy)
28 – position
32 – scroll x direction (–1,0,1 – left, none, right)
36 – scroll y direction (–1,0,1 – down, none, up)

Once you have extracted the scroll direction, the application can simulate the desired scroll by setting the scroll bar position (scx,scy) as descibed in last month's article. In order for this condition to occur, the scroll_request flag should be set when creating the window. (See 'Get_Window_Info', offset 28.)

The other conditions will be described in later articles as and when they become relevant. So don't worry! Here is a general method of implementing this:

```
DIM block% &100
amountx=10 : REM scroll amounts
amounty=10
:
mask=FNmask(...variables...)
:
REM main program
:
REPEAT
SYS "Wimp_Poll",mask,block% TO
reason
PROCaction(reason,block%)
UNTIL FALSE
END
:
DEFFNmask(...variables...)
LOCAL mask%
mask%=0
IF null THEN mask%=mask% OR &001
IF redraw THEN mask%=mask% OR &002
IF leave THEN mask%=mask% OR &010
IF enter THEN mask%=mask% OR &020
IF mouse THEN mask%=mask% OR &040
IF key THEN mask%=mask% OR &100
=mask
:
DEFPROCaction(reason,block%)
CASE action OF
WHEN 0 : REM go off and do
application processing
WHEN 2 : SYS
"Wimp_OpenWindow",,block%
WHEN 3 : SYS
"Wimp_CloseWindow",,block%
WHEN 4 : REM pointer left window
WHEN 5 : REM pointer entered window
WHEN 7 : REM user drag has finish
WHEN 10 : PROCscroll_acknowledged
ENDCASE
:
DEFPROCscroll_acknowlegded
scx=block%!20
scy=block%!24
scx+=(block%!32)*amountx : REM add
scroll amount in appropriate
direction
scy+=(block%!36)*amounty
block%!20=scx
block%!24=scy
SYS "Wimp_OpenWindow",,block% : REM
cause AWM to display new portion
ENDPROC
```

As an aside I thought that this would be an appropriate place in which to include the next three routines. They allow the user to obtain and set information on the windows.

'Get_Window_Info' at &400CC

This routine returns all the information on the required window in a data block (whose address the application must stipulate). The data block returned follows the 'Create_Window' format:

0 – initial work area coordinates (x0,y0,x1,y1) – 4 words

16 – scroll bar positions (scx,scy) – 2 words

24 – handle to open window behind (–1 top, –2 bottom)

28 – flags:–

&0001 has title bar
&0002 is moveable
&0004 has vertical scroll bar
&0008 has horizontal scroll bar
&0010 can be redrawn entirely by the wimp
&0020 is a 'pane'
&0040 is allowed to outside main area
&0080 has no 'back' or 'quit' boxes
&0100 'Scroll request' returned – auto-repeat
&0200 'Scroll request' returned – debounced

status:–

&0001 0000 window is open
&0002 0000 is on top
&0004 0000 has been toggled to full size

32 – colours (title foreground,background; work area foreground, background) – 4 bytes

36 – colours (scroll bar outer/inner) – 2 bytes

38 – colour of title background if the window has the input focus (i.e.highlight colour) – 1 byte

39 – reserved

40 – work area extent (mx0,my0,mx1,my1) – 4 words

56 – title flags:–

&0001 contains text
&0004 has a border
&0008 text is horizontally centered in box
&0010 text is vertically centered in box
&0020 has a filled background
&0040 text is anti–aliased font
&0200 text is right justified
&x000 0000
(x is the font number if text is anti–aliased)

60 – work area flags (default 'button type')

&x000 when x =
&0 – ignore mouse clicks
&1 – notify application when the pointer is over the window
&2 – click –> notify application (auto-repeat)
&3 – click –> notify application (debounced)
&4 – click –> select, release –> notify application (or deselect if moved away)

&5 – click –> select (double click) –> notify application
&6 – as (3) but can also drag (returns button state *16)
&7 – as (4) but can also drag (returns button state *16)
&8 – as (5) but can also drag (returns button state *16)
&9 – select when pointer is over window (notify if clicked)
&A – click –> buttons*256, drag –> buttons*16, double click –> buttons*1
&B–&E – reserved
&F – writeable window i.e. a mouse click causes caret to be positioned inside window
60 – reserved
72 – title string (12 bytes)
84 – number of icons in initial definition
88 – icon definitions (32 bytes each)

Don't worry if most of this doesn't make sense. I haven't covered all of it yet! I just included it at this stage for completeness and so that it could be used for reference at a later date. This would save you hunting through countless issues trying to find a relevant information.

Typical program:

```
DIM block% &100
:
PROCget_window_info(handle)
:
DEFPROCget_window_info(handle)
!block%=handle
SYS "Wimp_GetWindowInfo",,block%
:
REM remove data from block eg
flags=block%!28
:
ENDPROC
```

'Get_Window_State' at &400CB

This routine returns a shorten version of the above. The data block has the same layout as above but only includes data up to offset 28. This routine is designed to allow the application to access the transient variables of a window (i.e. the variables that change).

```
DIM block% &100
:
PROCget_window_state(handle)
:
DEFPROCget_window_state(handle)
!block%=handle%
SYS "Wimp_GetWindowState",,block%
:
REM remove data from block
:
ENDPROC
```

'Set_Extent' at &400D7

This routine allows the application to change the size of the work area extent (mx0,my0, mx1,my1). It requires, as input, the handle of the window and the new coordinates. The new area may not be set so that any part of the visible portion is excluded. Thus this call cannot cause the size of the visible window to be changed nor can it cause it to be scrolled.

```
DIM block% &100
:
PROCchange_extent(handle,mx0,my0,
                             mx1,my1)
END
:
DEFPROCchange_extent(handle,mx0,
                         my0,mx1,my1)
block!0=mx0
block%!4=my0
block%!8=mx1
block%!12=my1
SYS "Wimp_SetExtent",handle,
                              block%
ENDPROC
```

Now you should not only be able to display windows on the screen but also manipulate them. Next month I will show you how to make use of the windows, that is actually 'write' some data into them!

The example programs and procedures I write are just that: examples. They are essential reading, as they show how the WIMP enviroment should be implemented. The WIMP enviroment is very flexible, and because of this it requires a lot of inputs. This flexiblity is what makes it seem so complex.

When you actually come round to creating your own environment you will find that most of these inputs will either be global (set up only once) or not even required. For this purpose, each month, I write a program incorporating the relevant routines in the kind of environment you youselves may wish to use. Believe me, it is much easier to do it than it is to explain it! This example 'working' program is available on the Archive program disk. So if you want to see how its done why not buy a copy?! A

(Ed. He said that, not me, honest!)

RS423 Serial Port – The Last Word?!

This information was taken from an Acorn Applications Note on the subject.

The RS 232 specification defines the interface between Data Terminal Equipment (DTE) and Data Communication Equipment (DCE) employing serial binary data. In the case of an Archimedes using a modem, the computer is the DTE and the modem is the DCE. When a serial printer or another computer is connected to the Archimedes, both devices appear as DTE's and the interconnections will be different.

The RS 232 specification is very broad-based in its scope, and defines the use of a 25 way connector with many connections between the DTE and DCE. In practice, many equipment manufacturers have found that only a small sub-set of these connections is necessary and connectors with far fewer pins have been used. Unfortunately, the selection of the sub-set is often arbitrary and there may be confusion over how to make the best use of the connections.

Only two wires (plus ground) are theoretically needed to make a two-way serial data connection between the DTE and the DCE. The additional connections are primarily used to carry out "hardware handshaking". The handshaking enables the two pieces of equipment to exert some direct control over each other's data flow. This is necessary to avoid data loss when the receiving equipment is too busy to receive more data. Increasingly "software handshaking" is used instead of "hardware handshaking". In this case, whilst data is transmitted along one data line, the other line is carrying control information that allows the receiver to control the flow of data being sent.

Archimedes Serial Port

The Archimedes Serial Port uses a 9-way connector (D-type plug) that is similar in function, but is not identical, to the 9-way port on the IBM PC/AT or derivatives. The same connecting lead can normally be used as that supplied for the PC/AT. The pin connections for the 9-way connector are given opposite.

Connection to another computer

In this case, both devices behave as DTE's and the RXD and TXD lines need to be transposed between the computers. Software handshaking will often be used, with some hardware control "looped-back" locally:

Connect pins 1, 4 & 6 (DCD, DTR & DSR) together at each end.

Connect pin 2 (RXD) of the DTE (Archimedes)

to TXD of the DTE (computer)
Connect pin 3 (TXD) of the DTE (Archimedes) to RXD of the DTE (computer)
Connect pin 5 (0V) of the DTE to 0V of the DTE
Connect pin 7 (CTS) of computer A to pin 8 (RTS) of computer B
Connect pin 8 (RTS) of computer A to pin 7 (CTS) of computer B

Connection to a modem

In this case, the computer is the DTE and the modem is the DCE. The RXD and TXD lines are not transposed and the DCD, DTR & DSR lines may be brought into use. The TXD, RXD, CTS, RTS, DCD, DTR, DSR and 0V lines of one are connected to the same pins on the other.

Note: In both of the above arrangements, the CTS line on the Archimedes does not behave in quite the same way as for the existing Master Series or Model B computers. The CTS line on Archimedes when disabled will cause an immediate cessation of data flow. For byte-oriented protocols, this may be undesirable as the last character may be corrupted. For applications where this is likely to be a problem, the DSR input should be used instead of the CTS (i.e. the functions should be reversed). The DSR input, when disabled, allows completion of the character.

Connection to a Serial Printer

Both devices will typically be DTE's and some hardware handshaking may be needed. Connections will be similar to the "Computer-to-computer" method, but will vary between printer types. The printer manual should be consulted for guidance.

Baud Rate

The serial controller chip within the Archimedes provides one internal timer for baud rate control. If the receive and transmit rates are programmed to be the same, then this timer is used for both. If different baud rates are needed for transmit and receive, then the internal timer is allocated to transmit while the receive is allocated a system timer. This system timer is not an exact multiple or sub-multiple of the internal timer. The use of different receive and transmit rates is therefore only recommeded for lower speed usage e.g. 1200/75 Viewdata. If only receive is being used on Archimedes, it is important to program transmit to the same rate, even though it is not being used. This ensures that Arthur allocates the internal timer to receive.

Versions of Arthur

Early versions of Arthur, i.e. those prior to Version 1.00, have a problem which can affect received data. If it becomes necessary for Arthur to signal to a sender to stop sending data, due to a full buffer etc then Arthur fails to re-enable the sender when the problem clears. A patch is supplied on the Welcome disk for machines supplied with Arthur version 0.3. Customers with versions earlier than 1.2 should be receiving an upgrade in December 1987. A

RS423 Pin Connections

Pin	Function		Voltage & Logic state
1	DCD	Data Carry Detect	+ve = On (Dectected)
2	RXD	Receive Data	+ve = Logic"0"
3	TXD	Transmit Data	+ve = Logic "1"
4	DTR	Data Terminal Ready	+ve = On (Ready)
5	GND	Ground Return	(0V)
6	DSR	Data Set Ready	+ve = On (Ready)
7	RTS	Request To Send	+ve = On (Send)
8	CTS	Clear To Send	+ve = On (Clear)
9	RI	Ringing Indicator	+ve = On (Ringing)

Library, Machine Code or Module?

A 4.5" by 3.5" Archimedes screen dump for Epson printers by Gerrald Fitton

The subject of this series of articles is a printer dump for Epson printers, but one of the main aims is to illustrate various features of programming on the Archimedes. This month, Gerrald explains first, in more detail, about the self-contained BASIC function which was given last month and gives one or two possible modifications.

One of the valid criticisms of earlier versions of BASIC was that they encouraged unstructured, "spaghetti" programming. Unless they were very simple, these unstructured programs were very difficult to read, debug, modify, improve or extend. In most cases, the work put into writing one program could not be transferred into the next. Writing programs as self-contained parts, small enough to be understood by themselves, is more than an aid to writing extended programs, it is essential. Even the earliest BBC BASIC included functions and procedures. The ability to pass parameters and declare variables as local to the function or procedure lends itself to good structuring.

One of the many new exiting features of BASIC V is its LIBRARY and INSTALL commands which positively encourage the use of self-contained functions and procedures. The Archimedes has the capacity for large programs. One way of developing large programs to take advantage of this increased memory is to build up a library of well-documented, self-contained functions and procedures. These can then be loaded into memory and called from a fairly simple, short main program.

Functions and Procedures in BASIC V

A function or procedure is self-contained if the only variables in it are either passed to it as parameters or are declared as local from within the procedure. In other words no global variables are allowed within the function or procedure. One problem with the earlier versions of BBC BASIC was that arrays could not be declared from within a procedure and so could not be made local variables. This restriction does not apply to BASIC V. At line 50080 of last month's program the local variable vdu% is declared. On the next line a byte array is declared reserving 40 bytes in memory at the location pointed to by vdu%. This memory can be used without ambiguity in FNminidump. Another array of a different size called vdu% could have been declared and had values assigned to it in a procedure which calls FNminidump and yet, after leaving FNminidump those old values would still be preserved.

Parameters

The parameter margin% is the left margin in tenths of an inch. The parameter threshold% is the logical colour of a screen pixel which, if this value is exceeded, then the pixel is mapped to a printed dot.

Calls to the Operating System

One operating system call is used at line 50210 in FNminidump. This call uses a parameter block to read a selection of the VDU variables in a memory independent manner. At lines 50120 to 50200 the space reserved by the local byte array vdu% (line 50090) is used to set up the input parameter block. The numbers entered into the array select variables which return the screen graphics window (set up by VDU 24) and the graphics origin. Although not essential, the same space at vdu% is used for the values returned by the call and read into local variables such as xorigin% at lines 50220 to 50290. The variable xconvert% is 1 in screen mode 0, 2 in mode 1 and 3 in mode 2 indicating that there are 2, 2 squared and 2 cubed graphics co-ordinates per pixel in these modes respectively. This value of xconvert% has to be used to convert the values returned by right% etc since they are returned by the OS call as pixels, not graphics coordinates!

The Graphics Window

If a graphics window has been declared with VDU 24, or if there has been a change of origin, then the values of left% etc. will not be the usual (0,0) and (1279,1023) of the default screen. Lines 50320 to 50350 carry out the necessary conversion from the values returned to the parameter block by the OS call of line 50210. In line 50380 the number of strikes of the printer head is calculated.

Scanning the Screen

The screen is scanned in a maximum of 32 horizontal rows. Each row consists of a column of 8 pixels, this is 32 graphics co-ordinate units for all screen modes. The rows are scanned from left to right producing one printer strike per two graphics co-ordinate units. The variable line% is the value of the y co-ordinate of the top one of a column of 8 pixels. At line 50530 the BASIC function POINT(x%,y%) is used to read the logical colour of the pixel at graphics co-ordinates (x%,y%). If the value of colour% is greater than the value of the parameter threshold% then, in the next line, the appropriate amount to cause a dot to be printed, is added to the value of printcode%, the printer stike code which is sent to the printer at line 50560.

Printer Mode 5

To convert the program for the RX80 or FX80 printer change the three lines:-

```
50380 points%=(right%-left%+1)DIV4
50490 VDU 27,42,5,points%MOD&100,
                          points%DIV&100
50500 FOR x%=left% TO right% STEP 4
```

Use of Function

As Dr. Andrew Bangham pointed out in the Vol.1 No.1 issue of Archive, the command EVAL(dump$) is powerful but under-used. Different dumps can be called by the command dump%=EVAL(dump$) where, in our case, the variable dump$ is assigned the value `dump$="FNminidump("+STR$(margin%)+","+STR$(threshold%)+")"` but could be some other dump just as easily.

Self-contained Function Revisited (1)

At line 50580 of last month's fully self-contained function you will find the BASIC command VDU 10,13 which is a line feed, <LF>, followed by a carriage return, <CR>. This will operate correctly with most computer-printer setups. If you are getting either no line feeds or double line feeds, then read on.

The Epson printers are "bidirectional logic seeking" one consequence of which is that ASCII code 13, <CR>, does nothing. For historical reasons and for compatibility with earlier systems it is possible to configure the Epson to interpret ASCII 13 as a line feed. The handbook refers to a DIP switch which has the function "Automatic Line Feed ON/OFF". If this is ON then ASCII 13 will cause a line feed. The BBC Model B has an osbyte call *FX 6 for which the default is *FX 6,10. The default prevents ASCII 10 being sent to the printer. Hence for the BBC B system in its default state, the Epson switch must be ON.

Your choice with the Archimedes is either to configure the computer not to send ASCII 10 using *CONFIGURE IGNORE 10 <RETURN> and to set the DIP switch ON, or, alternatively *CONFIGURE IGNORE <RETURN> with the DIP switch OFF. Check the computer with *STATUS. My preference is to have a status of "No Ignore" with the DIP switch OFF because the printer is then compatible with IBM systems. The screen dump HardCopyFX of the Welcome disc sends only ASCII 10 (without 13) so to use it, set up "No Ignore" (and set the DIP switch OFF).

Self-Contained Function Revisited (2)

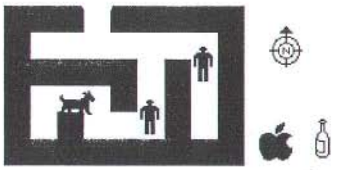
For a few rare combinations of ORIGIN and VDU 24 values the dump given last month fails. I have traced the fault to line 50380 which should be altered to points%=(right%-left%)DIV2+1 or to points%=(right%-left%)DIV4+1 for printer graphics mode 7 or 5 respectively. I changed line 50330 to overcome a problem and forgot about line 50380. What can I do but apologise? Sorry!

Next month we will look at the fully relocatable code version.

Minotaur by Minerva Systems

The first game to be published for the Archimedes was Minotaur. This is a fairly straight-forward maze game based on the legend of Theseus. You have to move around inside a 3-D maze rescuing men before they get eaten by the dreaded minotaur and trying to stay alive yourself. Things you may find on your journey around the maze are bits of map which give you a wider and wider overview of the maze, a compass which make the map considerably easier to interpret, a sword to help you if you happen to meet the minotaur and various cans of food and bottles of drink. As you move around, your food and drink gradually drain away until you die of hunger (or thirst, presumably, though it never happened to me).

Part of the fun of minotaur is that there are virtually no instructions, so you are groping round in the dark trying to find out what to do. I only recently discovered that you could use the mouse instead of the cursor keys to specify which direction you want to move – you click on one of the four arrows around the Minerva Systems logo. There are also one or two cryptic



comments in the instructions about magic buttons and sending messages to Ariadne for help, but I haven't found out what that is all about yet!

The program starts off with some very impressive introductory screens but the maze itself is only very basically drawn and the only sound is the familiar Archimedian bong – this occurs when you walk into a wall. The only thing for me which gets me to go on trying it is that I suspect there are other aspects that I have not yet discovered. It would be more gripping if it gave you some sort of score instead of telling you each time that you have died of hunger. All you can do is remember how many of the men you saved before you died.

Is it worth £14.95? (or £13 to Archive members?) Well, it's not cheap, but I suppose on the basis of supply and demand it can command that sort of price since, as I write, it's either that or Zarch at £19.95.

Sounds Interesting

Matthew Treagus (who is, I gather, almost 15!) gives an over-view of the Archimedes' sound system and lists the various commands available. This article only gives a starting point for examining this huge subject. There is so much to learn about the Archimedes' sound system!

The Archimedes seems to have unbelivable capabilities in all areas of computing and the sound system is no exception. The technical specifications of the system are superior to those of the famous rivals the Atari ST and the Amiga. The way the sound system interacts with BASIC and machine code is via Acorn's new SWI's (Software Interrupts). In this article I will attempt to provide some basic insight into this powerful area of Archie. I do not pretend to be the next Mozart or Bach but I will try to explain the computer side of the sound chips.

The sound system is "built up" and accessed at different levels. Normally, as programmers, we need only use the top level from BASIC unless we want Archie to speak, reproduce digital samples or sing! Occasionally we may wish to create our own voice and then we will have to use a lower level of access.

Level 2 –	Sound Event Scheduler
Level 1 –	Sound Channel Interface
Level 0 –	Sound DMA Buffer Handler
	Hardware

The Sound DMA Buffer Handle is the lowest level of sound system that handles the comings and goings of the data through the hardware.
The Sound Channel Interface handles the stereo channels, sample rates and sound system physical channels.
The Sound Event Scheduler contains the up-front note handling which will be of most use to the ordinary user.

The entire sound system can be accessed via the SWI calls that have been meticulously arranged by Acorn. These will be of great advantage to all machine code programmers who have bad memories of complex sound parameter blocks on previous BBC's. Instead of giving a complex and in–depth study of every command I will list the commands and explain the more important ones. For the duration of this article I will give examples in BASIC, for simplicity.

LEVEL 0 – Sound DMA Buffer Handler

Right at the root of the sound system are the sort of commands you would expect.

SOUND ON / SOUND OFF

This turns the sound system on or off.

VOICES <n> where <n> = 1 to 8

This command sets the number of sound channels available to the sound system. There is a maximum of 8 channels and either 1, 2, 4 or 8 channels can be selected for use. i.e. selecting VOICES 3 will actually allocate 4 voices.

STEREO <channel>,<position>
where <channel> = 1 to voices available, and <position> = –127 to 127

This command is used to set the stereo position of a sound channel rather like a balance knob on a stereo hi-fi only each channel can be at one of 7 different channel positions. This is of little use with the mono speaker on the Archimedes itself! –127 is maximum to the left and 127 is maximum to the right.

***AUDIO ON / OFF**

Turning Audio Off silences the sound system completely, as SOUND OFF in BASIC.

***SPEAKER ON / OFF**

This command affects the internal speaker only and not the external stereo output. It has the effect of silencing the mixed signal to the internal speaker.

***STEREO <channel> <position>**

As STEREO in BASIC.

LEVEL 1 – Sound Channel Handler

VOICE <channel>,<voicename>

where <channel> is the sound channel number (range 1–8), and <voicename> is a string to match a loaded voice generator.

The command attempts to match the named voice and install it against the channel specified. The voices available can be displayed by using *VOICES.

SOUND <channel>,<amplitude>, <pitch>,<duration>

<channel> is the sound channel (1–8)
<amplitude> is the peak not amplitude
<pitch> is the fundamental note pitch
<duration> in 1/20 seconds before note off

The cosmetics of this command are unchanged from the earlier Acorn machines but its effects are different. When the sound command is issued it makes the note defined in the parameters into the current note on that channel and rather than putting it to the end of the queue. Extensions to the amplitude and pitch parameters are described later.

***VOLUME <1–127>**

This command scales the voice generator sound tables to alter the overall loudness of the sound produced on all channels.

***VOICES**

This command lists the current voice generators and the channels they are allocated to (The Channel Allocation Map). A voice may be attributed to many different channels depending how many are in use. This can be set by VOICES (not to be confused with *VOICES) in Level 0.

***CHANNELVOICE <channel> <voice index> / <voice name>**

This command is used to allocate the different voice generators to the different channels. The voices are listed by *VOICES. The channel must be between 1 and 8 and the voice index must be a number. If the voice name is used it must be typed exactly as shown by *VOICES, and that includes the case of the letters.

***SOUND <channel> <amplitude> <pitch> <duration>**

This is the same as the BASIC command, SOUND.

***PITCH <1–32767>**

All pitch references are taken from the pitch base. This command is used to set the pitch base – all pitches for the voice generators vary with the system pitch base.

***CONFIGURE SoundDefault <0/1> <0–7> <1–16>**

Sets the Sound default in the CMOS memory. The first parameter controls the internal speaker. 1 for on, 0 for off. The second is the course volume of the sound system. The final parameter controls the voice to be allocated to channel 1 on start up. The volume of the bell sound is set with:

***CONFIGURE LOUD**
***CONFIGURE QUIET**

LEVEL 2 – Sound Event Scheduler

This is the simplest and most immediately useful part of the sound system and therefore the commands in this section are documented more fully than those above.

TEMPO <expression>

This sets the system TEMPO which allows fine control over a large range; the parameter is normally regarded as a 4 digit hexadecimal number. i.e, &wxyz. This number is a "hexadecimal fraction", the xyz part being the fraction and the w the whole part. Therefore &1000, the default value, represents 1 and 0/4096 and &2800 represents 2 and 1/2. The TEMPO is measured in microbeats/centisconds and so the value of &4000 sets the tempo to 4 microbeats/centisecond. The current TEMPO setting is returned in the variable TEMPO, i.e. you can say, for example, PRINT TEMPO.

BEATS <expression>

This command sets the microbeat counter limit. When the counter reaches this limit it resets to 0. All sounds are queued to this point. The default value is zero causing sounds to be played as soon as they are scheduled. Setting BEATS to a positive value causes the bar length to change at the end of the current bar. Setting BEATS to a negative value (less than –1) causes the microbeat counter to be disabled. –1 and 0 have no effect. The current BEATS setting is returned in the variable BEATS.

BEAT

The current BEAT is returned in the variable BEAT.

SOUND <channel>, <amplitude>, <pitch>, <duration>, <nBeats>

<channel> is the sound channel (1–8)
<amplitude> is the peak not amplitude
<pitch> fundamental note pitch
<duration> in 1/20 seconds before note off
<schedule period> is the time in microbeats relative to the last bar start

This command treats the first four parameters exactly the same as those in the Level 1 SOUND command but the sound specifed is queued on that channel for the specified number of beats after the bar start when BEAT = 0

***TEMPO <n>**

This is the same as the BASIC command, TEMPO.

***QSOUND <channel> <amplitude> <pitch> <duration> <nBeats>**

This queues in a sound after the specified number of nBeats.

Glossary

Amplitude – The volume or loudness of a note. Measured from –15 to 0 or on a logarithmic scale from 256 to 511.

Centi–seconds – One hundredths of a second.

Channel – A stack on which notes are stored and passed down prior to being played.

Micro Beat – One tick of the beat counter.

Note – A combination of channel, amplitude, voice, pitch and duration and in some cases Q times.

Pitch – The frequency of a note given as a number from 0 – 255 or between 256–32767.

SWI – (Software Interrupts) should be available from most future languages and are neat and tidy ways of getting into Arthur (the Operating System)

Q times – The amount of time ticks needed before the next note is played.

Voice – Exactly the same as a voice generator.

Voice Generator – A routine that accesses lower levels of the sound system to produce a sound – usually to recreate the sound of another instrument. New Voice Generators are loaded into the system as Relocatable Modules (RM's) using the *RMLOAD. These then appear on the Channel Allocation Map as displayed by *VOICES. Two are provided on the Welcome Disk in the modules directory, they are "Percussion" and "StringLib". They each contain 4 voices.

Volume – See Amplitude

Sound System SWI's

Here is a list of the SWI's associated with the sound system.

Level 0	SWI "Sound_Configure"
	SWI "Sound_Enable"
	SWI "Sound_Speaker"
	SWI "Sound_Stereo"
Level 1	SWI "Sound_InstallVoice"
	SWI "Sound_RemoveVoice"
	SWI "Sound_AttachNamedVoice"
	SWI "Sound_AttachVoice"
	SWI "Sound_Volume"
	SWI "Sound_SoundLog"
	SWI "Sound_LogScale"
	SWI "Sound_Pitch"
	SWI "Sound_Tuning"
	SWI "Sound_Control"
	SWI "Sound_ControlPacked"
	SWI "Sound_ControlBlock"
	SWI "Sound_WriteControlBlock"
Level 2	SWI "Sound_QInit"
	SWI "Sound_QSchedule"
	SWI "Sound_QRemove"
	SWI "Sound_QFree"
	SWI "Sound_QDispatch"
	SWI "Sound_QTempo"
	SWI "Sound_QBeat"

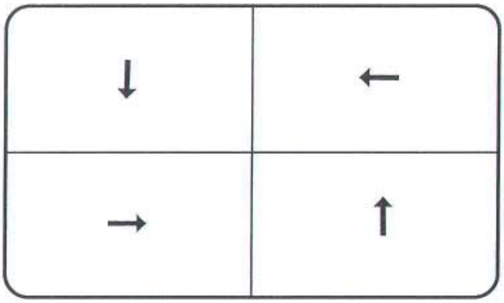
Further details of these calls can be found in the "A Series Programmer's Reference Guide" which should be available by the time this article is printed. This Guide will also contain information for anyone who wants to write their own voice generators.

```
 10 REM >RNDSCROLL
 20 REM (C) W.R.HINDLE Oct 1987
 30 MODE9:W%=40:C%=7
 40 VDU19,0,24,0,240,240
 50 OFF
 60 MOUSE ON
 70 *POINTER
 80 REPEAT
 90 TIME=0:REPEAT UNTIL TIME>5
100 MOUSE X%,Y%,B%
110 GCOL 1,B%
120 ELLIPSE FILL X%,Y%,RND(99)+40,RND(70)+40
130 A%+=1:IF A%>C% A%=1
140 GCOL 0,A%
150 VDU28,0,15,W%/2-1,0
160 VDU23,7,0,2|
170 RECTANGLEFILL RND(500),RND(200)+712,RND(99)+40,RND(72)+40
```

Sideways Scrolling with VDU23

Rob Hindle

Here is a program which demonstrates the use of the new VDU23,7 command to scroll windows in any of four directions. The program splits the screen into four text windows, draws graphics objects in each and scrolls each window in a different direction.



The following program notes should help you to see how it works.

30 Select an appropriate screen mode, set screen width (in characters) variable W% and maximum colours C% excludes black and flashing colours).

40 Create a coloured border around the screen

50-70 Turn off cursor, activate mouse & pointer

80 Start main loop

90 Delay - without this, the program runs a little too fast to see what is happening. Run the program once, then delete this line to see the speed improvement.

100-120 Draw an ellipse of random size at the current mouse location in a colour determined by which of the mouse buttons are pressed. This is more effective when the mouse points to somewhere near the edge of the screen. Pressing two or all three of the buttons produces more colours.

130-140 Cycle through the available colours

150 Select a text window, in this case the top left quadrant of the screen

160 Scroll the current text window, in this case, downwards by one row

170 Draw a random size graphics object within the current text window using the colour selected at lines 130,140. In this case, the object is a filled rectangle.

180-200, 210-230, 240-260 Repeat the operations done in lines 150-170 but for that other three quadrants of the screen, scroll in a different direction and draw a different graphics object.

Note the use of W%>>1 which is a one bit shift operation and is equivalent to in integer divide (i.e.W%/2) but marginally faster.

```
180 VDU28,0,31,W%/2-1,16
190 VDU23,7|
200 RECTANGLE RND(300),RND(400),RND(99)+40,RND(72)+40
210 VDU28,W%>>1,15,W%-1,0
220 VDU23,7,0,1|
230 ELLIPSE RND(240)+940,RND(310)+612,RND(60)+40,RND(60)+40
240 VDU28,W%>>1,31,W%-1,16
250 VDU23,7,0,3|
260 ELLIPSE FILL RND(440)+740,RND(210)+100,RND(60)+40,RND(60)+40
270 UNTIL0                                                   ,RND(45)
```

A Beginner's Guide to the ADFS

from material provided by **M J Hobart, Paul Jolly & James Lynn**

I asked last month for someone to write an article about the ADFS and ended up with four separate articles! I have combined three of them into this one article and kept back a fourth (rather more technical) article for the next issue.

No matter what work you do with your computer, whether it is word-processing, spreadsheet work or programming, you will always need to preserve your work when you have finished. The standard method of storage on the Archimedes is 3.5" discs. But a disc is useless without a sensible way of arranging the information on the disc. This is the job of the Advanced Disc Filing System (ADFS). To draw an analogy with a filing cabinet, this too is used to store and retrieve information. We can look at the cabinet as similar to a floppy disc, and the equivalent to the ADFS might be a filing clerk, who knows how the information is arranged in the cabinet, and can retrieve the information you are interested in at any particular time.

What's the difference?

Some of you may be familiar with the old DFS system, so it may help to compare to the two. The two main differences between the old DFS and ADFS are:

(1) ADFS supports a herarchical structure in which the 'root' ($) directory can contain both files and subdirectories. The sub-directories can, in turn contain both files and subdirectories, more or less ad nauseam.

(2) ADFS treats both sides of a disk as if they were one.

The first benefit is that you can store as many files as you like, since, although each directory can only contain 47 objects (files or subdirectories), the number of subdirectories is not limited. The second is that files can live in a sensibly named directory: e.g. a letter to someone to whom one often writes (Fred) might be called by the date on which it was sent, living in the subdirectory called after the person, which is itself a subdirectory of subdirectory 'LETTERS', which lives in subdirectory 'WP'. The full name of the file might thus be $.WP.LETTERS.Fred.87Sept19.

Starting from scratch

Now for some practicalities. To start using a brand new disc it must first of all be formatted. This is necessary because a blank disc has no information telling the ADFS where it can put data. A formatted disc is divided up into a number of concentric rings, called tracks, which are in turn divided into sectors. The number of tracks and sectors and the size of each sector determines how much information can be stored on a disc. Tthe ADFS command, *FORMAT can be used to format a disc in one of two different ways. The first gives you 640K of space on the disc and is compatible with the ADFS on the BBC micro. The second format, only available on the Archimedes, gives 800K free on a disc and also makes disc accessing somewhat faster. The actual syntax of the *FORMAT command is *FORMAT <drive> [L/D] where <drive> is the drive number, L means a 640K format and D means an 800K format. Once the disc is formatted you can save and load information as you require.

Handling files

The ADFS gives many different commands to handle the files on a disc. The *DELETE command allows you to delete a specific file from the disc, e.g. *DELETE letter will delete the file 'letter' from the current directory. You can change the name of any file on the disc using the *RENAME command, e.g. *RENAME first second will change the name of the file called 'first' to 'second'.

You can also copy files from one disc to another, using the *COPY command. The full syntax is *COPY <source> <destination> [C][D][F][P][Q][R][V] where <source> is the name of the file to copy, and <destination> is the destination filename. The rest of the parameters are optional. [C], if present, will ask for confirmation before each copy. [D] will delete the source file after copying, effectively making it a move command. [F] will force the COPY command to overwrite any files with the same name as the destination filename. [P] will prompt the user to swap discs, allowing you to copy a file from one disc to another using only one drive. [Q] tells the ADFS that it can use application workspace when copying the files (see later). [R] means 'Recurse', meaning any sub-directories on the disc you are copying will also be copied. [V] means verbose, information is displayed about each file when copied.

To explain the [Q] option further, what it means is that if you are not using the application workspace for anything else (for instance, running BASIC) then the ADFS can use it to copy large files. If this option is not used and the file to be copied is particularly large, then the ADFS will give a "No Room" error. This occurs because the ADFS seems to have to load a complete file into memory when copying and normally it only has a small area of memory in which to load the files. With the Q option, it can use the whole memory of the Archimedes to load and save files.

Another command which uses this option is *BACKUP which copies one whole disc onto another. When using it, it is much quicker to use the [Q] option since, on a 1MByte machine, it can usually copy a whole 800K disc with only one disc swap. However, you must be careful when using this option, at least with Arthur 0.20, since BASIC has a tendency to crash if its workspace is overwritten with a [Q] command. The best course of action is to leave whatever application you are using. (The command QUIT will exit BASIC, *GOS will exit from other applications into the supervisor.)

Using only these commands, you can quite happily get along using the ADFS, but there will no doubt come a time when you might want to organise the information on your discs in a more coherent fashion – after all, with a maximum of 77 filenames possible in a disc directory, sometimes you will want a way of arranging related files so they are grouped together. The ADFS lets you do this by arranging files into different directories. A directory is effectively a file which holds a list of other files on the disc.

Treating the catalogue as just another file allows the ADFS to increase the number of possible files on disc from 47 (on a 640K disc) or 77 (on an 800K disc) to as many as you want. This is because, although the original directory on the disc (usually referred to as the 'root directory') has a limit on the total number of filenames it can list any of the files in the directory can be another directory (or sub-directory) with the same number of possible entries and in turn, any file in that sub-directory can be a further sub-directory.

This type of arrangement is called a 'hierarchical structure'. A sensible use of a hierarchical filing system can prove very beneficial, although much of the time, when using floppy discs, it will probably only be of marginal benefit. However, it is often useful to partition off different files in different directories. For instance, on a disc you might have one directory called 'WORDS' for all your word processing and another called 'FIGURES' for your spreadsheet work and perhaps a third called 'LIBRARY' for useful programs you often need when working. We will use this example to see how to use the ADFS to create directories. On a fresh disc, to create a new directory in the root directory, you use the command *CDIR <name>. In our example we would use the three commands *CDIR WORDS, *CDIR FIGURES and *CDIR LIBRARY. Once you have created a directory, the easiest way to use it is to make it the current directory, using *DIR <dir name>.

A different view the ADFS

Think of the root directory as a cupboard with a door marked ROOT or simply $. Opening the door allows you access to what's inside. In this cupboard are several smaller cupboards, each with its own door. On each door there is a name. The name on a door can be either a file name or it can be a catalogue name. Opening a door with a catalogue name reveals another set of cupboards with names on the doors and again, the door names can be either a filename or the name of a catalogue.

If you can let your imagination run even further, pretend that you are capable of climbing into the cupboards. Now the hard part – we want to find a file called '$.magazine.archive.fact_file'.

1. We open the cupboard (that is *MOUNT) – remember that the root directory is '$' –
2. we look for the cupboard marked 'magazine'
3. now open the cupboard marked 'magazine' and step in (i.e. *DIR magazine)
4. we look in the cupboard for the door that is marked 'archive'.
5. opening the door and stepping into that cupboard (equivalent to *dir archive) we see on one of the doors facing us the characters 'fact_file'.
6. behind this door is the file we seek!

When we executed step 3 and were within the cupboard marked 'magazine' it was impossible to read the writing on any other doors within the first cupboard. This is true of ADFS, when you are in a directory *CAT just gives you a catalogue of the directory you are in. To obtain access to, or information about, another file or directory we need to know its 'path name' – this is the name you would use when trying to get to it from the ROOT directory (cupboard). Thus a file could be called from step 6 above, by using its pathname '$.magazine.beebug.help'. Another method is to back out of the cupboards until we find the common cupboard (in this case 'magazine'), and using the character ^ to back out of the cupboard. (This is termed return to the parent directory.) We can write '^.beebug.help' and this will take us from the directory 'archive' back to directory 'magazine' then into directory 'beebug' and finally to the file 'help'.

Careful planning

Now a word of warning… Even with a floppy disc, but especially with a winchester (or hard) disc, you absolutely **must** think before you leap. In other words your structure must relate to a thought pattern. It is no use placing programs of similar languages, types, or related files into random positions in the directory structure. They will be lost forever.

Leave your computer switched off, walk away from it and get out a pencil and paper, sit down in a quiet corner and plan out your structure. This may seem silly at first but in due course, you will reap the benefit of the time you have invested when, for example, you need a program in a hurry. If you are using floppy discs then assume that, for the purposes of planning, the root directory contains all the floppy discs as directories and mark them with the names you have given them. The process of installing them onto a hard disc at a later date is simplified and because you have been thinking of the discs themselves as part of the directory structure for a relatively long time, you will have no trouble with the installed structure on the winchester.

A good habit to absorb is placing the pathname for a program on the first line of that program as a remark. e.g.:-

```
10 REM > $.basic.analysis.S_parameter
```

for BASIC, or for pascal:-

```
{$.pascal.analysis.S_parameter }
```

The main reason for placing the pathname on the first line is that of traceability since from a printout you will be able to see exactly where the program is stored and how to get there again. But in the case of the BASIC program, the added ">" character means that you can just type SAVE without using the filename.

Accessing files

If you need to access a file in another directory, simply the file name will not do, as the ADFS will try and find the file in the current directory. What you need to give is the path that the ADFS will have to take to find that file. Let's say you are in the root directory on a disc, and you have to access a file called LETTER in a directory called WORDS which is in the current directory. The pathname of the file is made up of the directory that holds the file, and the filename itself, separated by a full stop, i.e. WORDS.LETTER. If the file happened to be in a lower level of directory, for instance in a sub-directory of WORDS called STUFF, the pathname would be WORDS.STUFF.LETTER and so on down as many levels as you wish to go. This method is most often used when the file needed is in a lower level directory to the one you are in, but it is dangerous to use it all the time, as the pathname changes depending on which is the current directory. A more general way of expressing the path of a file is by including the root directory name in the pathname. The root directory on any ADFS disc is called $. In our last example, the full pathname of the file would be $.WORDS.STUFF .LETTER. This pathname could be used regardless of the current directory.

Another useful shortcut in a pathname is the ^ symbol, which refers to the directory above the current directory. For instance, if the current directory was STUFF in WORDS, and you gave the command *DIR ^, the current directory would become WORDS. If we now wanted to access a file called TAKINGS in the directory FIGURES we could use the pathname ^.FIGURES.TAKINGS. This seems a waste since $.FIGURES.TAKINGS is exactly the same length, but the ^ example would still find the correct file, no matter in which directory the directories FIGURES and WORDS are located.

A third symbol used in the ADFS is &. This is referred to as the 'user root directory' and normally has exactly the same meaning as the root directory. However, using the *URD <name> command you can set the user root directory to be a different directory. This is probably of most use with a winchester disc, since it would probably be used when running application programs which all have their own specific data files.

Many application programs need to access data files of some sort, and they usually require them to be in specific directories. If all these programs kept accessing their files via the root directory $, then the root directory would suddenly get very full up with datafiles and subdirectories. It would be far more sensible to have separate directories somewhere on the disc for each application, and before running those applications set the URD to the top level for that particular application. Then, as long as that application uses & instead of $, all data and other related files could be found just as if there was no other information on the disc.

As an example, suppose a particular program always wanted to find information in a directory called DATA. If the program used a filename such as $.DATA.FILE1, when transferring the program to a winchester disc, you would have to ensure that the directory DATA always appeared in the root directory. However, if the program looked for &.DATA.FILE1, the user could specify in which directory the DATA directory was sitting, thus making the arrangement of files on the disc much easier.

Another facet of a pathname is the disc name. Usually, this is simply the number of the drive where the disc is located. When used in a pathname, the number is prefixed by a colon (:) as in :0.FILE meaning a file called FILE in drive 0. A disc can also have a specific name, which is set using the *NAMEDISC command. For example, *NAMEDISC 0 WORK will name the disc in drive 0, 'WORK'. From now on, if you use the name in a pathname (for example :WORK.FILE), if the appropriate disc is not present, the error message 'Disc not present' will be given. In this way, you can make sure that you do not get the wrong information just because the wrong disc happens to be in the drive and it happens to have a file of the same name on it. A

ADFS Hints and Tips

• If you want to catalogue a drive other than the currently mounted drive, you can type, e.g. *CAT:1 or just *.:1 but you will sometimes find that if you try it on an external 5.25" drive you get the error "Drive empty" whether or not that is true. The reason for this is that the 5.25" drives do not generally have a "disc inserted" control line. (This is another reason why Acorn say that you should not add an external 5.25" drive.)

Similarly, you can catalogue a directory without actually going into it. For example, if you put in the Welcome Disc and *MOUNT it and then type *CAT MOD* and you will get a catalogue of the MODULES directory.

• If you are trying to COPY and get messages about not having enough space, QUIT to the operating system and try again.

• Since you can specify the language a particular file should run under, you no longer need to have a !BOOT file saying *BASIC and CHAIN "PROG". Instead you can create a BASIC program called !BOOT and instead of a *OPT4,3 you do *OPT4,2 to *RUN the !BOOT file. If you are using the desktop then you would just click on the !BOOT file and run it.

• *WIPE * Acorn have obviously realised the dangers of using *WIPE* that I mentioned in the first issue of Archive. Apparently in the 0.3 operating system this has been changed so that the default is to give you prompts unless you tell it otherwise – that sounds much more user-friendly!

• The *SETTYPE command can be used to give a file a different identity. For example, if you have ported a BASIC program across from a BBC machine as a file and you do a *INFO on it, it will not say that it is a BASIC program, instead, it will have something like &F0E in that column. If you *SETTYPE PROG &FFB (assuming the program is called PROG) it will then respond as a BASIC program when you do *INFO on it. The types that we know of are:

&FFF	Text
&FFE	Command
&FFD	Data
&FFC	Utility
&FFB	Basic
&FFA	Module
&FF9	Sprite
&FF8	Absolute code - runs as an application at &8000
&FF7	BBC Font
&FF6	Fancy Font
&FEF	Diary
&FEE	Notepad
&FED	Palette
&FE0	Desktop utility

Apart from the obvious advantage of being able to look at the *INFO list and see quickly the identity of all the different files in a particular directory, it also means that you can run the program directly from a * command. From any language (I think) you can say *PROG and, provided the file PROG is in the current directory, the relevant language, perhaps BASIC, will be activated and the program will be CHAINed. It should therefore be theoretically be possible, to have a suite of programs which call each other and which are not necessarily written in the same language. Being lazy, I would be tempted to *PROG a particular program. The only disadvantage is that when you escape from the program it drops you back into Arthur, so you have to select BASIC and OLD if you want to edit the program.

• If you name your discs by using something like *namedisc :0 FRED you can then put statements into programs that refer to the disc name when you are trying to load or save information. Thus instead of saying something like X%=OPENIN ":0.$.DATA" you could say X%=OPENIN":FRED.$.DATA" so that if it did not find the requested disc in the current drive, it would go through all the others until it found the disc specified.

• Some people find it convenient to type directory names in capitals and files in lower case. Most of my work is wordprocessing and I keep format, heading, temporary and signature files in the root directory. They can be accessed from anywhere, most conveniently by programming the function keys to do the work.

• More about *MOUNT – This important command tells the ADFS that you have changed the disc in the drive. It is necessary since the ADFS always 'remembers' what the current directory is, and if you swap discs, it will usually tell you 'disc changed'. It is sometimes useful not to have to keep typing *MOUNT every time you want to swap discs. Fortunately, a *CONFIGURE option exists to make sure the ADFS reads in a new catalogue every time you swap discs.

To use it, type *CONFIGURE NODIR, press <return> and then press <ctrl-break> to set the option. From now on, whenever you swap discs, the ADFS will read in a new directory from the disc you have just inserted. This does not work, however, if you enter a directory on the disc other than the root directory because now the ADFS has remembered a directory, and will get confused if you swap discs. From then on, it will require a *MOUNT command for the ADFS to recognise a new disc, or a *NODIR command to tell the ADFS to forget the directory it is currently holding.

SWI Listing Program

Neil Strong

The following program is a small utility which displays the string names for the complete set of SWI's installed by the operating system, podules, modules etc. It uses the SWI "OS_SWINumberToString" function to search for all currently available SWI's. It produces about twenty or thirty pages of information but if you want to stop it, you can press <escape>. If you want the listing sent to the printer, just adding an extra line at the beginning: 5 VDU2 will switch the printer on and line 690 switches it off again in any case. Note that lines 740 and 750 consist of strings made up of lots of spaces, not the actual words "9 spaces" or "50 spaces". **A**

```
 10 REM (C)Neil Strong, 11.11.87
 20 DIM code% 1000
 30 FOR pass%=0 TO 2 STEP 2
 40 P%=code%
 50 pc=15
 60 link=14
 70 sp=13
 80 notswi=12
 90 swicount=11
100 chkswi=10
110 notxswi=9
120 :
130 [OPT pass%
140 STMFD (sp)!,{link}
150 ADR notswi,notused           ; "User" SWI to ignore
160 LDR notswi,[notswi]
170 ADR notxswi,notused+4        ; "XUser" SWI to ignore
180 LDR notxswi,[notxswi]
190 MOV swicount,#0              ; Start at 0
200 .nextswi
210 MOV R0,swicount              ; set up params to translate
220 MOV R2,#50                   ; max SWI string length
230 ADR R1,buffer                ; place string here
240 SWI "OS_SWINumberToString" ; and convert
250 LDR chkswi,[R1]              ; get first 4 chars from string
260 CMP chkswi,notswi            ; if "User" then don't print
270 BLEQ skip
280 CMP chkswi,notxswi           ; if "XUse" then don't print
290 BLEQ skip
300 ;
310 SWI "OS_WriteS"
320 EQUS "SWI = &":EQUB 0          ; print "SWI = &< SWI number>"
330 MOV R0,swicount
340 ADR R1,hexbuf
350 MOV R2,#9
360 SWI "OS_ConvertHex8"
370 ADR R0,hexbuf
380 SWI "OS_Write0"
390 SWI "OS_WriteS"                ; print 3 spaces
400 EQUS "   "
410 EQUB 0
420 ADR R0,buffer                  ; and SWI name
430 SWI "OS_Write0"
440 SWI "OS_NewLine"               ; CR/LF
450 ;
460 .skip
470 BL chkesc                      ; Check for escape key
480 ADD swicount,swicount,#1       ; Next SWI
490 CMP swicount,#&FF000000        ; check to see if max reached yet
510 LDMFD (sp)!,{pc}               ; else return to BASIC
520 ;
530 .chkesc
540 MOV R0,#129                    ; OSBYTE read key
550 MOV R1,#0
560 MOV R2,#0
570 SWI "OS_Byte"
580 CMP R1,#27                     ; is it an escape?
590 MOVNE pc,link                  ; No? then ignore it
600 LDMFD (sp)!,{pc}               ; else return to basic
610 ;
620 .notused EQUS "User"
630          EQUS "XUse"
640 .buffer  EQUS "               50 Spaces                        "
650 .hexbuf  EQUS " 9 Spaces"
660 ]
670 NEXT
680 CALL code%
690 VDU3
```

Order Form

		Quantity	Total
File Transfer Service:	Send your 5.25" discs, DFS or ADFS and £5 per disc, inclusive, and we will transfer the files onto an ADFS 3.5" disc. (One 3.5" for each 5.25".)		
Program Disc	Vol 1.1 – £2 / Vol 1.2 – £3 / Vol 1.3 – £3		
Blank Keystrips	5 blank keystrips on A4 card – £1 per card		
Unformatted 3.5" Discs	Bulk pack, Wabash – 10 for £16		
	Wabash in library cases – 10 for £18		
Wordwise Plus – Full Package	£30		
Wordwise Plus – Upgrade	£10		
(Please quote registration number.)			
CP-ROM on Disc – Full Package	£18		
CP-ROM on Disc – Up-Grade	£6 (Please return old CP-ROM)		
"ARM Assembly Language Programming"	£12		
Minerva Deltabase	£26		
Minerva System Delta-Plus	£64		
Minerva Minotaur	£13		
Clares' Archimedes Toolkit Module – £37			
Clares' Imagewriter – £27			
Clares' Artisan Art Package – £37			
Programmers Reference Manual	£28		
Shareware Graphics Demonstration Disc	£3		

I enclose a cheque payable to "Norwich Computer Services" for:
(Sorry on Acces or Visa facilities.)

All prices are inclusive of VAT and UK carriage.

Name ______________________________

Address ______________________________

____

____

Fact-File

Brainsoft	22 Baker Street, London, W1M 1DF.
CJE Micros	Dept AM, 78 Brighton Road, Worthing, W Sussex, BN11 2EN. (0903-213361)
Clares Micro Supplies	98 Middlewich Road, Rudheath, Northwich, Cheshire, CW9 7DA. (0606-48511)
Computer Concepts	Gaddesden Place, Hemel Hempstead, Herts, HP2 6EX. (0442-63933)
Contex Computing	15 Woodlands Close, Cople, Bedford, MK44 3UE. (02303-347)
EMR Ltd	14 Mount Close, Wickford, Essex, SS11 8HG. (0702 - 335747)
Fairhurst Instruments	Dean Court, Woodford Road, Wilmslow, SK9 2LT. (0625 525 694)
HS Software	56, Hendrefolian Avenue, Sketty, Swansea, SA2 7NB. (0792-204519)
Intelligent Interfaces	14 Julius Close, Chandlers Ford, Eastleigh, Hants, SO5 2AB. (04125-61514)
Meadow Computers	11, London Street, Whitchurch, Hants, RG28 7LH. (025689-2008)
Minerva Systems	69 Sidwell Street, Exeter, EX4 6PH. (0392 - 37756)
Pineapple Software	39 Brownlea Gardens, Seven Kings, Ilford, Essex, IG3 9NL. (01-599-1476)
RESOURCE	Exeter Road, Doncaster, DN2 4PY. (0302-63800/63784)
Serious Statistical Software	Lynwood, Benty Heath Lane, Willaston, South Wirral, L64 1SD. (051 327 4268)
Tubelink	P.O.Box 641, London, NW9 8TF.

Archive

The Subscription Magazine for *Archimedes* users

Articles in previous issues:

- File transfer on RS423
- Attaching a 5.25" drive
- C.C.'s ROM-Link packages
- Clares' Toolkit module
- Graphics demo programs
- Archie the Music Computer
- Structuring with BASIC V
- Technical notes on BASIC V
- Epson Screen Dump
- Using BBC ROM images

Future issues will include:
BBC micro as an I/O podule, using windows, using the operating system, ARM assembly language programming, writing relocatable modules, etc etc.

Technical Help Service (£8 / year)
A telephone hot-line service for immediate help with your technical problems. Any member can send written enquiries, but for a fast response use the THS!

Members discount: 7.5% off software from Computer Concepts, Minerva Systems and Clares Micros Supplies purchased through Norwich Computer Services.

Subscription: 12 issues £10 (UK/Eire) Europe £16, Middle East £20, America / Africa £23, Elsewhere £25. Technical Help Service £8

* Please send copies of *Archive* magazine for one year starting from
Vol. 1, Nº 1, Oct '87 / Vol. 1, Nº 2, Nov '87 / Vol. 1, Nº 3, Dec '87.

* Please enrol me on the Technical Help Service for one year. (£8)

I enclose a cheque for £ ________

Name: ____________________

Address: ____________________

____

____________________ Postcode: ________

Norwich Computer Services, 18 Mile End Road, Norwich, NR4 7QY.
Subscription – £10 per year, Technical Help Service – £8 per year